Cascaded All-Optical Shared-Memory Architecture Packet Switches Using Channel Grouping Under Bursty Traffic

A Dissertation

Presented to

The Academic Faculty

by

Michael D. Shell

In Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy in Electrical and Computer Engineering

School of Electrical and Computer Engineering

Georgia Institute of Technology

November, 2004

Cascaded All-Optical Shared-Memory Architecture Packet Switches Using Channel Grouping Under Bursty Traffic

Approved by:

Joseph L. A. Hughes, Advisor
School of Electrical and Computer Engineering

W. Russell Callen
School of Electrical and Computer Engineering

John A. Copeland
School of Electrical and Computer Engineering

Richard M. Fujimoto
College of Computing

Henry L. Owen
School of Electrical and Computer Engineering

Date Approved: November 17, 2004

To my mom and dad

Acknowledgments

The quest for a doctoral degree in the hard sciences or engineering is not for the faint of heart. Only those who undertake such an endeavor can fully appreciate the magnitude of the ordeal which demands endurance as well as ability.

First and foremost in my life has been the unwavering love and support of my parents, Columbus and Ingeburg Shell. All my best qualities are in large part attributable to them, and any of my faults are entirely my own. If everyone in the world would be as fortunate with regard to parents as I am, our most pressing problems would vanish in short order.

My advisor, Dr. Joseph L. A. Hughes, is also deserving of credit. He not only provided me with an excellent topic, but he also patiently allowed me both the freedom and the time needed to bring this work into full fruition. I consider it an honor to have worked for him. Dr. W. Russell Callen, Dr. John A. Copeland, Dr. Richard M. Fujimoto and Dr. Henry L. Owen graciously agreed to serve on the various committees and review this work. It is professors such as these that make Georgia Tech the great school that it is. I have also benefited over the years from the advice of Dr. Dale C. Ray and Dr. Roger P. Webb. On a sad note, I will dearly miss Dr. Daniel C. Fielder who passed away in October 2002 just a few days short of his 85th birthday. Many generations of students have benefited from his unique insight, dedication, wisdom and kindness. Professor Fielder taught me combinatorics which is so prominent in this work.

I am also grateful to Dr. Peter D. Bergstrom Jr., my predecessor with regard to work on this topic. Peter's pioneering work provided an excellent starting point for my own thesis. Dr. Shih-Cheng (Tony) Wang, a good friend of mine since my early graduate days, was kind enough to provide me with his LaTeX dissertation typesetting files, which were based on an earlier style file from Dr. Paul Anderson, at a time before Georgia Tech officially supported such a thing. Since then, I have learned a lot about,

and contributed some to, LaTeX. I would also like to thank Janet Myrick and Bob House for their help in the earlier phase of my research with administrative work and prototype construction, respectively, as well as their advice and friendship.

Finally, I wish to take the somewhat unusual step of thanking a few people whom I have never had the opportunity to meet. Dr. Donald E. Knuth's development and subsequent free release of TeX has benefited scientific and technical writers for more than a quarter of a century. I now thoroughly enjoy using TeX to typeset my research papers. Furthermore, Knuth created the Computer Modern font used in this TeX-produced document. I chose it because of its formalism, completeness and unrivaled ability to produce beautiful mathematical symbols. Knuth is also one of the (if not *the*) foremost experts in the theory of the generation of pseudo-random number sequences. Some of his algorithms were used in the coding of my network simulator (SONSIM). I would also like to thank Linus Torvalds for his creation of the Linux kernel and Richard Stallman of the Free Software Foundation for the development of the GNU utilities, compilers, and development tools which are an integral part of the Linux operating system. Linux is a real joy to work with, and I have benefited greatly from having a powerful, stable, secure, open and reliable computation and application development platform with the rich heritage of Unix — all for free on my desktop.

We live in the most remarkable of times. Never before has human technology reached such a high state of development, let alone coupled with increasingly rapid advancement. I believe that in this century, quite possibly within my lifetime, we will obtain the answers to our most profound and ancient questions. Can the emperor reign over his own success? I await the future with a strange mix of hopeful optimism, excitement, trepidation, awe and wonder.

mds

Contents

Acknowledgments . . . . . **iv**

List of Tables . . . . . **x**

List of Figures . . . . . **xi**

Nomenclature . . . . . **xv**

Summary . . . . . **xxi**

Chapter 1: Introduction and Background **1**

1.1 Introduction 1

1.1.1 Purpose and Contributions of Work 2

1.2 ESMP Switches 3

1.2.1 Starlite and Sunshine Switches 5

1.3 OSMA Switches 5

1.4 Non-OSMA Switches 9

1.4.1 Karol's Shared-Memory Optical Packet (SMOP) Switch 9

1.4.2 Haas' Staggering Switch 11

1.5 Channel Grouping 11

1.6 Banyan Networks 13

1.7 Markov Models 14

1.7.1 The Use of Markov Chains to Analyze Systems 15

1.7.2 Existence of the Steady-State Probabilities 16

Chapter 2: Prior Work **18**

2.1 Karol and Hluchyj 18

2.2 Szymanski and Shaikh 18

2.3 Liew and Lu 19

2.4 Lin and Silvester 19

2.5 Izmailov and Haas 19
2.6 Turner and Bianchi 20
2.7 Pattavina, Monterosso and Gianatti 20
2.8 Montagna, Paglino and Meyer 21
2.9 Fong and Singh 21
2.10 Saleh and Atiquzzaman 22
2.11 Danielsen 22
2.12 Bergstrom 22
2.13 Chia and Hunter 23
2.14 Singh, Kushwaha and Bose 23
Chapter 3: OSMA Switches Under Random Traffic 25
3.1 The Uniform Random Source (RS) 25
3.1.1 The Extended Random Source (ERS) 26
3.2 State Descriptions 27
3.2.1 General Buffer Description 28
3.2.2 Buffer Cells are Indistinct 28
3.2.3 Destination Addresses are Indistinct 29
3.3 Calculation of the Transition Probabilities 31
3.3.1 The OSMA Operation Cycle 32
3.3.2 Nonreduced Arrival Vectors 36
3.3.2.1 Probabilities of Nonreduced Arrival Vectors from RS . 36
3.3.2.2 Probabilities of Nonreduced Arrival Vectors from ERS 36
3.3.3 Arrival Vector Reduction (AVR) 37
3.3.3.1 Probabilities of Reduced Arrival Vectors 39
3.4 Calculation of Switch Performance 39
3.4.1 Loss Probability and Throughput 39

3.4.2 Buffering and Direct Routing Probabilities 41
3.4.3 Expected Number of Packets in the Buffer 42
3.4.4 Expected Packet Delay Times 42
3.5 Numerical Results 43
Chapter 4: OSMA Switches Under Bursty Traffic 50
4.1 The Two-State Source (TS) 50
4.1.1 The Extended Two-State Source (ETS) 52
4.1.2 The Temporal-Burstiness Factor 54
4.2 Spatial Burstiness 54
4.2.1 The Spatial-Burstiness Factor 55
4.3 System States Under Bursty Traffic 56
4.3.1 General Input Source State Description 57
4.3.1.1 General Input Source State Transition Probabilities . . 57
4.3.2 Reduced Input Source State Description 58
4.3.2.1 Reduced Input Source State Transition Probabilities . 58
4.4 System State Transition Probabilities 60
4.4.1 $\mathcal{P}_\alpha$ With Each Channel Driven Independently 60
4.4.2 $\mathcal{P}_\alpha$ For Sped-Up TS Source Driven Links 60
4.4.3 $\mathcal{P}_\alpha$ Under ETS Traffic 61
4.5 Calculation of Switch Performance 61
4.6 Model Algorithm Overview 62
4.7 Scalability Issues 63
4.8 Numerical Results 65
4.8.1 The Effect of Burstiness on the Packet Loss Rate 67
4.8.2 The Effect of Load on the Packet Loss Rate 69
4.8.3 The Effect of Burst Length on the Packet Loss Rate 70

4.8.4 Reducing the Packet Loss Rate via Channel Grouping and/or Increasing the Buffer Size . . . 71

Chapter 5: Networks of OSMA Switches . . . 75

5.1 Interstage Traffic Approximation . . . 75

5.2 Output Traffic of OSMA Switches . . . 77

5.2.1 Interstage Load and Temporal Burstiness . . . 77

5.2.2 Interstage Spatial Burstiness . . . 80

5.2.3 Eigentraffic . . . 82

5.3 Determining the Virtual Source Model . . . 82

5.3.1 Output Traffic Information Required from the Switch Model . . . 83

5.3.2 Matching Obtained Traffic Parameters to an ETS Source . . . 85

5.3.3 Factors That Contribute to Inexactness . . . 87

5.4 Overall Loss Rate of a Network . . . 88

5.5 Numerical Results . . . 89

Chapter 6: Asymmetric Switches . . . 96

6.1 Trill Networks . . . 98

6.1.1 Numerical Results . . . 100

Chapter 7: Conclusion . . . 104

7.1 Contributions of Work . . . 105

7.2 Suggestions for Future Research . . . 105

Appendix A: Partitions of Integers . . . 108

Appendix B: Author Publications . . . 117

References . . . 119

Vita . . . 126

List of Tables

Table 1 Selected data points for the traffic parameters used in Figure 26. . . 69

Table 2 Interstage traffic properties at the output of each stage for a 16×16 Banyan network of 2×2 bufferless switches, $c = 4$, under a random traffic load of 0.30. 80

Table 3 Performance, as determined by the analytic model, of $c = 1$, 128×128 Trill networks with expansion ratios of 1, 2 and 4, under a random traffic load of $\sigma = 0.30$. In each of the three cases, there is a total of 832 buffer cells in the network. 103

List of Figures

Figure 1 The ESMP switch. 3

Figure 2 The all-optical Starlite OSMA switch. 6

Figure 3 The multiwavelength fiber loop memory OSMA switch. 7

Figure 4 OSMA versus ESMP loss performance for 4×4 switches with 16 buffer cells under random traffic. ESMP data taken from Pattavina and Gianatti's exact vectorial model [24, page 406]. 8

Figure 5 Karol's shared-memory optical packet (SMOP) switch. 10

Figure 6 Haas' Staggering switch. 11

Figure 7 A switch with channel grouping on the outputs. 12

Figure 8 An 8×8 Baseline Banyan network using 2×2 switches. 14

Figure 9 OSMA switch parameters (all-optical Starlite version shown). . . . 25

Figure 10 The random source. 26

Figure 11 Interfacing a random source to a multichannel link. 26

Figure 12 The extended random source 27

Figure 13 $\mathcal{P}_{\text{loss}}$ for $n = 4$ switches under random traffic with $p = 0.5$. 44

Figure 14 $\mathcal{E}_b$ for $n = 4$ switches under random traffic with $p = 0.5$. 44

Figure 15 $\mathcal{P}_{\text{loss}}$ for $n = 4$ switches under random traffic with $p = \frac{0.5}{c}$. 46

Figure 16 $\mathcal{P}_{\text{loss}}$ for $n = 4$ switches under random traffic with $p = 0.5$, $s = 1$, $r = 1, 2$. 47

Figure 17 $\mathcal{P}_{\text{loss}}$ for $n = 4$, $c = 1$ switches under random traffic with $p = 0.5$, $\frac{0.5}{2}$, $\frac{0.5}{4}$. 47

Figure 18 $\mathcal{P}_{\text{loss}}$ for $m = n$, $c = 1$ switches under random traffic of various loads. 48

Figure 19 The two-state source. 50

Figure 20 Interfacing a two-state source to a multichannel link. 52

Figure 21 Temporal versus spatial burstiness. 55

Figure 22 Number of system states required for the model of this work ("partition" model) compared to that of Pattavina's "3-D" model [24]. 64

Figure 23 $\mathcal{P}_{\text{loss}}$ for $n = 8$ switches under ETS traffic, $\lambda = 8,\ p_c = 1.0,\ \sigma = 0.3$, ($\beta \approx 2.9$ and $\beta_S = 1, 2.533$ for $c = 1, 4$, respectively). 65

Figure 24 $\mathcal{P}_{\text{loss}}$ for $n = 8$, $c = 1$ switches under random (RS) and bursty (TS) traffic. 66

Figure 25 Effect of β on $\mathcal{P}_{\text{loss}}$ for an $n = 8$, $c = 1$, $m = 8$ switch. Two-state source (TS) parameters are β, $p = 0.6$, $\sigma = 0.3$. $\mathcal{P}_{\text{loss}}$ under $p = 0.3,\ 0.6$ random traffic (RS) is also shown for comparison. . . 67

Figure 26 Effect of β_S on $\mathcal{P}_{\text{loss}}$ for an $n = 4$, $c = 4$, $m = 8$ switch under ERS traffic with $\sigma = 0.3$. 68

Figure 27 Effect of σ on $\mathcal{P}_{\text{loss}}$ for $n = 2, 8$, $c = 1$, $m = n$ switches under bursty traffic of $\lambda = 4$, $p = 1.0$, $(0.9 < \beta \leq 7.5)$. 69

Figure 28 Effect of σ on $\mathcal{P}_{\text{loss}}$ for $n = 2, 8$, $c = 1$, $m = n$ switches under bursty traffic of $\beta = 1.5$, $\lambda = 4$, $(0.2 \leq p \leq 1.0)$. 70

Figure 29 Effect of λ on $\mathcal{P}_{\text{loss}}$ for $n = 2, 8$, $m = n$ switches under bursty traffic of $\beta = 1.5$, $\sigma = 0.3$, $0.47 < p \leq 0.9$. 71

Figure 30 $\mathcal{P}_{\text{loss}}$ for $n = 2$, $c = 1, 2, 4$ switches under bursty traffic, $\lambda = \frac{8}{c}$, $p_c = 1$, $\sigma = 0.5$. 72

Figure 31 $\mathcal{P}_{\text{loss}}$ for $n = 4$, $c = 1, 2, 4$ switches under bursty traffic, $\lambda = \frac{8}{c}$, $p_c = 1$, $\sigma = 0.5$. 72

Figure 32 $\mathcal{P}_{\text{loss}}$ for $n = 2$, $c = 1, 2, 4$ switches under bursty traffic, $\lambda = \frac{8}{c}$, $p_c = 1$, $\sigma = \frac{0.5}{c}$. 73

Figure 33 $\mathcal{P}_{\text{loss}}$ for $n = 4$, $c = 1, 2, 4$ switches under bursty traffic, $\lambda = \frac{8}{c}$, $p_c = 1$, $\sigma = \frac{0.5}{c}$. 73

Figure 34 $\mathcal{P}_{\text{loss}}$ for $n = 4, s = 1, r = 1, 2$ switches under bursty traffic, $\lambda = 8$, $p = 1$, $\sigma = 0.5$. 74

Figure 35 $\mathcal{P}_{\text{loss}}$ for $n = 2, 4$, $c = 1$ switches with large buffers under bursty traffic, $\lambda = 8$, $p = 1.0$, $\sigma = 0.5$, (yielding $\beta = 1.75$). 74

Figure 36 Approximating interstage traffic with virtual sources. 76

Figure 37 A cascade of switches. 77

Figure 38 $\mathcal{P}_{\text{loss}}$ from simulation for a 16×16 Banyan network of 4×4 switches with $c = 1,\ 0 \leq m \leq 4$ under random traffic. 78

Figure 39 σ at the output of each stage as measured from simulation. 78

Figure 40 β at the output of each stage as measured from simulation. 79

Figure 41 $\frac{\mathcal{P}_{\text{out},c}}{\mathcal{P}_{\text{in},c}}$ versus n and c for bufferless switches under a random traffic load of 0.5. 81

Figure 42 A nonlinear response of $\mathcal{P}_{\text{loss}}$ to variations in traffic properties contributes to errors when a parameter average is used as an approximation. 87

Figure 43 $\mathcal{P}_{\text{loss}}$ for switches in each stage, from both the analytic model and simulation, for a 16×16 Banyan network of 2×2 switches with $c = 4$, $0 \leq m \leq 4$ under random traffic, $\sigma = 0.3$. 90

Figure 44 $\mathcal{P}_{\text{loss}}$ for switches in each stage, from both the analytic model and simulation, for a 16×16 Banyan network of 2×2 switches with $c = 4$, $0 \leq m \leq 4$ under FOB bursty traffic, $\beta = 2.5$, $\beta_S \approx 2.53$, $\lambda = 4$, $p_c = 1$, $\sigma = 0.3$. 91

Figure 45 $\mathcal{P}_{\text{loss}}$ of the overall 16×16 Banyan network under random and FOB bursty traffic, $\sigma = 0.3$. 92

Figure 46 $\mathcal{P}_{\text{loss}}$ for switches in each stage, from both the analytic model and simulation, for a 16×16 Banyan network of 2×2 switches with $c = 4$, $m = 4$ under moderately bursty traffic, $1 \leq \beta \leq 1.8$, $2 \leq \lambda \leq 10$, $\beta_S \approx 1.56$, binomial p_i (for $\sigma_{\text{on}} = 0.6$), $\sigma = 0.3$. 93

Figure 47 $\mathcal{P}_{\text{loss}}$ for the overall 16×16 Banyan network under moderately bursty traffic, $1 \leq \beta \leq 1.8$, $\beta_S \approx 1.56$, $\sigma = 0.3$. 94

Figure 48 $\mathcal{P}_{\text{loss}}$ for switches in each stage, from both the analytic model and simulation, for a 64×64 Banyan network of 8×8 switches with $c = 1$, $m = 10$ under bursty traffic of varying burst lengths, $\beta = 1.25$, $3 \leq \lambda \leq 12$, $\sigma = 0.5$. 94

Figure 49 $\mathcal{P}_{\text{loss}}$ for the overall 64×64 Banyan network under bursty traffic of varying burst lengths, $\beta = 1.25$, $3 \leq \lambda \leq 12$, $\sigma = 0.5$. 95

Figure 50 $\mathcal{P}_{\text{loss}}$ for $c = 1$, $g = 4$, $m = 12$ switches under random traffic of $\sigma = 0.5$ and $\sigma = \frac{0.5g}{h} = \frac{2}{h}$ as a function of h ($2 \leq h \leq 8$). 96

Figure 51 $\mathcal{P}_{\text{loss}}$ for $c = 1$, $h = 4$, $m = 12$ switches under random traffic of $\sigma = 0.5$ and $\sigma = \frac{0.5}{h/g} = 0.125g$ as a function of g ($2 \leq g \leq 8$). . . . 97

Figure 52 The Trill architecture. 99

Figure 53 $\mathcal{P}_{\text{loss}}$ for switches in each stage and the overall network, from both the analytic model and simulation, for a 16×16 Trill network of 2×4, 2×2 and 4×2 switches with $c = 4$, $m_1 = m_2 = 3$, and $m_3 = 4$ under moderately bursty traffic, $1 \leq \beta \leq 1.8$, $2 \leq \lambda \leq 10$, $\beta_S \approx 1.56$, binomial p_i (for $\sigma_{\text{on}} = 0.6$), $\sigma = 0.3$. 101

Nomenclature

$\lvert\ \rvert$	$\triangleq$	cardinality of a set, or the number of elements in a vector or matrix
$\forall$	$\triangleq$	"for all"
$\prime$	$\triangleq$	indicates a revised term
$\star$	$\triangleq$	indicates a term that differs in a controlled way
α	$\triangleq$	number of arriving packets
β	$\triangleq$	temporal-burstiness factor
β_S	$\triangleq$	spatial-burstiness factor
γ	$\triangleq$	periodicity of a Markov chain
η	$\triangleq$	number of stages in a network
λ	$\triangleq$	average burst length
$\boldsymbol{\pi}$	$\triangleq$	steady-state probabilities
π_i	$\triangleq$	steady-state probability for state i
$\boldsymbol{\pi}^t$	$\triangleq$	state probability distribution at time t
π_i^t	$\triangleq$	probability of a system being in state number i at time t
$\boldsymbol{\Psi}(k)$	$\triangleq$	set of unrestricted partitions of k
$\boldsymbol{\Psi}(k,\omega)$	$\triangleq$	set of partitions of k not exceeding ω in length
$\boldsymbol{\Psi}(k,\omega,\theta)$	$\triangleq$	set of partitions of k not exceeding ω in length with no element exceeding θ
$\boldsymbol{\Psi}_i(k)$	$\triangleq$	partition i
$\Psi_{i,j}(k)$	$\triangleq$	element j in partition i
σ	$\triangleq$	the normalized average load presented by a source
σ_{on}	$\triangleq$	the normalized average load presented by a bursty source when on
Υ	$\triangleq$	operator to produce a sum of products generated via selection
ϖ	$\triangleq$	number of arrival vectors represented by a reduced arrival vector
a	$\triangleq$	number of on-sources that remain on during an input state transition

AVR	$\triangleq$	arrival vector reduction
$\boldsymbol{\mathcal{A}}$	$\triangleq$	the set of arrival vectors
$\boldsymbol{\mathcal{A}}_y$	$\triangleq$	arrival vector y
$\mathcal{A}_{y,i}$	$\triangleq$	element i of arrival vector y
b	$\triangleq$	number of packets in the buffer
$\boldsymbol{\mathcal{B}}$	$\triangleq$	set of all buffer states
$\lvert\boldsymbol{\mathcal{B}}\rvert$	$\triangleq$	total number of buffer states
$\boldsymbol{\mathcal{B}}_x$	$\triangleq$	buffer state number x
$\boldsymbol{\mathcal{B}}'_x$	$\triangleq$	buffer state after routing from buffer
$\boldsymbol{\mathcal{B}}''_x$	$\triangleq$	buffer state after buffering contending incoming packets
$\mathcal{B}_{x,i}$	$\triangleq$	element i of buffer state number x
c	$\triangleq$	channel grouping factor of the links of a symmetric switch or of a given link
c_i	$\triangleq$	channel grouping factor of the links of a symmetric switch in stage i of a network
CGF	$\triangleq$	channel grouping factor
d	$\triangleq$	number of packets that need buffering
$\mathbf{D}$	$\triangleq$	packets that need buffering for each output link
D_i	$\triangleq$	number of packets that need buffering and are addressed to output link i
e	$\triangleq$	number empty buffer cells after routing out from buffer
ERS	$\triangleq$	extended random source
ESMP	$\triangleq$	electronic shared-memory packet switch
ETS	$\triangleq$	extended two-state source
$\mathcal{E}_\alpha$	$\triangleq$	expected number of packets arriving per timeslot
$\mathcal{E}_{\alpha\lvert x}$	$\triangleq$	expected number of packets arriving per timeslot, given that the system is in state number x
$\mathcal{E}_b$	$\triangleq$	expected number of packets in the buffer
$\mathcal{E}_{l\lvert x}$	$\triangleq$	expected number of packets lost in each timeslot, given that the system is in state number x

$\mathcal{E}_{\text{links},i\|x}$	$\triangleq$	expected number of output links that carry i packets, given that the system is in state number x
$\mathcal{E}_{t_\text{b}}$	$\triangleq$	expected time a buffered packet spends in the buffer
$\mathcal{E}_{t_\text{p}}$	$\triangleq$	expected time a packet takes to pass through the switch
$\mathcal{E}_{t_\text{s}}$	$\triangleq$	expected time a packet takes to be serviced by the switch
$\mathcal{E}_{w\|x}$	$\triangleq$	expected number of packets sent to the buffer in each timeslot, given that the system is in state number x
FOB	$\triangleq$	full on burst. Applies to any bursty source that, when on, always uses the full bandwidth of its driven link.
g	$\triangleq$	number of switch output channel groups (links)
g_i	$\triangleq$	number of output channel groups (links) for a switch in stage i of a network
$\mathbf{G}$	$\triangleq$	numbers of buffer state elements with common values (buffer groups)
G_i	$\triangleq$	number of buffer state elements with value i
$\mathbf{H}$	$\triangleq$	numbers chosen from each buffer group
H_i	$\triangleq$	number chosen from buffer group i
h	$\triangleq$	number of switch input channel groups (links)
h_i	$\triangleq$	number of input channel groups (links) for a switch in stage i of a network
I	$\triangleq$	total number of independent input sources
$\boldsymbol{\mathcal{I}}$	$\triangleq$	set of all input states
$\boldsymbol{\mathcal{I}}_{x_\mathcal{I}}$	$\triangleq$	input state number $x_\mathcal{I}$ when the input states are represented by vectors (general input source state description)
$\mathcal{I}_{x_\mathcal{I},i}$	$\triangleq$	element i of input state number $x_\mathcal{I}$
$\mathcal{I}_{x_\mathcal{I}}$	$\triangleq$	input state number $x_\mathcal{I}$ when the input states are represented by scalar values (reduced input source state description)
J_i	$\triangleq$	number of output links with i packets
K_i	$\triangleq$	number of elements of value i in a given partition or buffer state
l	$\triangleq$	number of packets lost in a transition
$\mathbf{L}$	$\triangleq$	contending packets that lost buffer access for each output link

L_i	$\triangleq$	number of contending packets that lost buffer access and are addressed to output link i
m	$\triangleq$	number of switch buffer cells
m_i	$\triangleq$	number of buffer cells for a switch in stage i of a network
n	$\triangleq$	number of input and output channel groups (links) for a symmetric switch
n_i	$\triangleq$	number of input and output channel groups (links) for a symmetric switch in stage i of a network
N	$\triangleq$	number of input and output links for a symmetric network
OSMA	$\triangleq$	optical shared-memory architecture
$\mathbf{O}$	$\triangleq$	number of packets at each output link after routing from the buffer
O_i	$\triangleq$	number of packets at output link i after routing from the buffer
$\mathbf{O'}$	$\triangleq$	number of packets at each output link after routing from the inputs
O'_i	$\triangleq$	number of packets at output link i after routing from the inputs
p	$\triangleq$	probability of a random (or on-state bursty) source emitting a packet in a timeslot
p_i	$\triangleq$	probability of a random (or on-state bursty) source emitting i packets in a timeslot
p_{on}	$\triangleq$	transition probability of a two-state source from off to on
p_{off}	$\triangleq$	transition probability of a two-state source from on to off
$\mathbf{P}$	$\triangleq$	transition probability matrix
$P_{i,j}$	$\triangleq$	probability of transitioning from system state number i to state number j
$\mathcal{P}_\alpha$	$\triangleq$	probability of α packets arriving
$\mathcal{P}_{\alpha \mid x_\mathcal{I}}$	$\triangleq$	probability of α packets arriving, given that the system is in input source state number $x_\mathcal{I}$
$\mathcal{P}_\delta$	$\triangleq$	probability of arrival vector destination addresses occurring
$\mathcal{P}_\mathcal{A}$	$\triangleq$	probability of a particular arrival vector occurring
$\mathcal{P}_{\text{buffer}}$	$\triangleq$	probability of an input packet being buffered
$\mathcal{P}_L$	$\triangleq$	probability of a particular $\mathbf{L}$ occurring

$\mathcal{P}_{\text{loss}}$	$\triangleq$	probability of an input packet being lost
$\mathcal{P}_{\text{loss},i}$	$\triangleq$	$\mathcal{P}_{\text{loss}}$ for a switch in stage number i of a network
$\mathcal{P}_{\text{in},i}$	$\triangleq$	probability that a tagged input link has i packets
$\mathcal{P}_{\text{out},i}$	$\triangleq$	probability that a tagged output link has i packets
$\mathcal{P}_{\text{out},i\|x}$	$\triangleq$	probability that a tagged output link has i packets, given that the system is in state number x
$\mathcal{P}_{\text{srcout}\|x_{\mathcal{I}}}$	$\triangleq$	probability that the input sources will emit at least one packet that is addressed to a tagged output link, given that the system is in input source state number $x_{\mathcal{I}}$
$\mathcal{P}_{x\|\text{out},i}$	$\triangleq$	probability of being in system state number x, given that the tagged output link has i packets
$\mathcal{P}_{\mathcal{T}}$	$\triangleq$	probability of a particular system state transition occurring
$\mathcal{P}_{\mathcal{T}_{\mathcal{B}}}$	$\triangleq$	probability of a particular buffer state transition occurring
$\mathcal{P}_{\mathcal{T}_{\mathcal{I}}}$	$\triangleq$	probability of a particular input state transition occurring
$\mathcal{P}_{\text{via}}$	$\triangleq$	probability of an input packet being directly routed to the output
q_i	$\triangleq$	probability that i packets are in a given link
r	$\triangleq$	channel grouping factor of the switch outputs
r_i	$\triangleq$	channel grouping factor of the outputs of a switch in stage i of a network
RS	$\triangleq$	random source (Bernoulli)
s	$\triangleq$	channel grouping factor of the switch inputs
s_i	$\triangleq$	channel grouping factor of the inputs of a switch in stage i of a network
SE	$\triangleq$	switching element, a switch within a network
SMELTER	$\triangleq$	Shared-optical-Memory switch Expected Loss calculaToR
SONSIM	$\triangleq$	Shared-Optical-memory switch Network SIMulator
$\boldsymbol{\mathcal{S}}$	$\triangleq$	set of all Markov states
$\|\boldsymbol{\mathcal{S}}\|$	$\triangleq$	total number of Markov states
$\boldsymbol{\mathcal{S}}_x$	$\triangleq$	system state number x
$\boldsymbol{\mathcal{S}}_x^{\mathcal{B}}$	$\triangleq$	the buffer-state part of system state number x

$\boldsymbol{\mathcal{S}}_x^{\mathcal{I}}$	$\triangleq$	the input-state part of system state number x when the input states are represented by vectors (general input source state description)
$\mathcal{S}_x^{\mathcal{I}}$	$\triangleq$	the input-state part of system state number x when the input states are represented by scalar values (reduced input source state description)
TS	$\triangleq$	two-state source
T_h	$\triangleq$	normalized throughput
$\mathbf{U}$	$\triangleq$	unused (free) channels in each output link after routing from the buffer
U_i	$\triangleq$	number of unused (free) channels in output link i after routing from the buffer
v	$\triangleq$	number of input packets routed directly to outputs
$\mathbf{V}$	$\triangleq$	input packets that are routed directly to each output link
V_i	$\triangleq$	number of input packets that are routed directly to output link i
VS	$\triangleq$	virtual source
w	$\triangleq$	number of packets sent to the buffer
$\mathbf{W}$	$\triangleq$	contending packets that won buffer space for each output link
W_i	$\triangleq$	number of contending packets that won buffer space and are addressed to output link i

Summary

This work develops an exact logical operation model to predict the performance of the all-optical shared-memory architecture (OSMA) class of packet switches and provides a means to obtain a reasonable approximation of OSMA switch performance within certain types of networks, including the Banyan family.

All-optical packet switches have the potential to far exceed the bandwidth capability of their current electronic counterparts. However, all-optical switching technology is currently not mature. Consequently, all-optical switch fabrics and buffers are more constrained in size and can cost several orders of magnitude more than those of electronic switches. The use of shared-memory buffers and/or links with multiple parallel channels (channel grouping) have been suggested as ways to maximize switch performance with buffers of limited size. However, analysis of shared-memory switches is far more difficult than for other commonly used buffering strategies. Obtaining packet loss performance by simulation is often not a viable alternative to modeling if low loss rates or large networks are encountered. Published models of electronic shared-memory packet switches (ESMP) have primarily involved approximate models to allow analysis of switches with a large number of ports and/or buffer cells. Because most ESMP models become inaccurate for small switches, and OSMA switches, unlike ESMP switches, do not buffer packets unless contention occurs, existing ESMP models cannot be applied to OSMA switches. Previous models of OSMA switches were confined to isolated (non-networked), symmetric OSMA switches using channel grouping under random traffic. This work is far more general in that it also encompasses OSMA switches that (1) are subjected to bursty traffic and/or with input links that have arbitrary occupancy probability distributions, (2) are interconnected to form a network and (3) are asymmetric.

Chapter 1

Introduction and Background

1.1 Introduction

The rapid growth of the internet has placed extreme demands upon the world's telecommunications infrastructure. This has resulted in the urgent need to develop new technologies that can deliver the required bandwidth. Techniques such as wavelength division multiplexing (WDM) can allow a single optical fiber to carry an impressive and seemingly unlimited data rate. Single-fiber data rates in excess of ten trillion bits per second have been reported [1], [2]. However, a serious bandwidth performance problem remains with the electronic switches that route data between the network links. A network switch must handle all of the data presented by the total of its input fibers and must be able to route this data, in real time, to the proper output ports. Furthermore, the incoming data must not be degraded or lost to an unacceptable degree in passing through the switch. Packet-switched networks place an additional constraint of high-speed routing operation upon the switch. This work focuses exclusively on packet-switched networks — specifically, time-slotted networks carrying fixed-length packets.

To improve the bandwidth capability of switches so they more closely match that of an optical fiber, the all-optical network [3]–[7] has been proposed. In an all-optical network, the data remain in an optical state from the original source, through the intermediate switches, until the final destination is reached. Although the switches may be electrically controlled, the packets they route remain entirely in optical form.

If these all-optical switches can operate at the needed packet rate with acceptable levels of signal degradation, a serious problem still remains — how to handle contention. Contention occurs when two or more packets compete for a limited network resource that cannot serve them all at once. Specifically, this work is concerned

with output port contention, which occurs when more packets arrive at a switch's inputs destined for a particular output than that output can handle at once. Buffering, the use of multichannel links and deflection routing are the most commonly proposed techniques for contention resolution in all-optical networks [8]. If the switch is unable to buffer packets or otherwise resolve the conflict, one or more packets will be lost. Packet loss rates of 10^{-10} or better are generally regarded as acceptable in today's ATM networks [9]. However, future all-optical networks may be more tolerant in this regard, especially if a much larger available bandwidth offsets the higher loss rates.

1.1.1 Purpose and Contributions of Work

It is the purpose of this work to develop a method of analysis to predict the packet loss performance of certain types of all-optical packet-switched networks that use the optical shared-memory architecture (OSMA) class of switches. The developed model yields exact results for an isolated switch or switches in the first stage of a network and good approximate results for switches in later network stages under random or bursty traffic. The following are all original contributions of the model in this work:

- It can be used to analyze OSMA switches under bursty traffic and/or with input links that have arbitrary occupancy probability distributions.
- It can be used to analyze Banyan networks with multiple stages of OSMA switches by using an interstage traffic approximation.
- It is general enough to be used with asymmetric (with respect to number of ports and/or channel grouping factors of the inputs and outputs) OSMA switches.
- It is more computationally tractable than the exact OSMA models previously reported in the literature [10]–[13].

1.2 ESMP Switches

In conventional fiber optic networks, packet switching is accomplished by converting the optical data stream into an electrical form via detectors, switching it electrically and then transmitting it via modulated lasers at the appropriate output port. An electronic buffer (memory) is provided to store packets that could not be routed because of contention. A switch in which a single memory bank is shared among all the ports is known as a shared-memory switch. For a given total number of switch buffer cells, shared-memory switches perform better than alternative buffer techniques such as input or output buffering (in which each port has its own buffer) because all the free memory cells can be used to hold packets from any port as needed. Furthermore, if there is a free output port, packets in the buffer destined to it can be routed out, regardless of their location within the buffer. Thus, the random access nature of the shared memory does not suffer from the head-of-line blocking problems exhibited by first-in-first-out (FIFO) memories used for input or output buffering. Electronic shared-memory packet switches (ESMP) (Figure 1) are sometimes referred to as very large scale integration (VLSI) switches because of the added switch complexity associated with allowing buffer cells to be shareable.

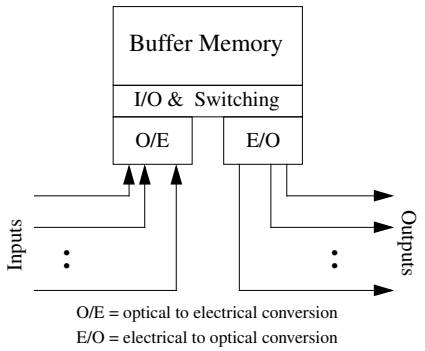


Figure 1: The ESMP switch.

It is helpful to review the steps in the logical operational cycle of an ESMP switch:

1. Packets arrive at the inputs, where the data is detected and converted into an electrical form. Addressing information is extracted at this time.

2. All of the packets are stored in the buffer. If there are more incoming packets than there are free buffer cells, packets are randomly selected to be stored. The excess packets are dropped.

3. As many packets as possible are routed from the buffer to the output ports. If there are more packets in the buffer destined for a particular output port than that port can handle, packets are randomly selected to be routed from the contending group. The unchosen packets remain in the buffer.

There are variations in the operational cycle. Most of these involve deviations from randomly selecting packets to be dropped or routed to accommodate packets with varying priorities, to limit the maximum time a packet remains in the buffer, or to preserve packet ordering. Giving certain packets priority does not affect the loss characteristics of a switch because the effect is simply to interchange packets with the same destination, but which differ in some other metric.

The ESMP switch offers a number of advantages. Because the optical signals are threshold detected and regenerated, noise does not accumulate from switch to switch. Also, because all packets that pass through the switch are buffered or latched, timing variations can be completely compensated for by the switch. Finally, the buffer can be made quite large because of the inexpensive nature of the memory. A large single-chip VLSI shared-memory switch can have 32 ports or more and a buffer size that exceeds a thousand packets [14]. Multiple-chip designs can be much larger.

There are also a number of disadvantages. Because the switch is electronic, it must provide optoelectronic interfaces, such as detectors and lasers, at the inputs

and outputs, respectively. These interfaces greatly increase the complexity and cost of the switch. Furthermore, the electronics must be able to process all of the data that passes through the switch. At the core of a network shared by many users, the presented bandwidth could be extremely large. Another issue is that the performance of the entire switch may be compromised if traffic imbalances cause a single output to monopolize the buffer. Various buffer management protocols have been proposed to minimize this problem [15].

It should be noted that that performance could be improved by allowing the ESMP switch to route packets directly from the input to the output and buffering only when contention occurs. This isn't often done in practice because it is much more cost effective to route all traffic through a larger buffer than to provide a "thru" bus which would complicate the routing circuitry and protocols.

1.2.1 Starlite and Sunshine Switches

One notable electronic shared-memory switch that supports "thru" capability is the Starlite switch [16]. However, in its unmodified form, the Starlite architecture is considered unfair as it favors dropping contending packets destined for higher address ports over those destined for the lower ports [17]. Techniques have been developed to deal with this issue [18].

The Sunshine switch [19] is a Starlite switch with added output buffers. Exact performance models for the Sunshine switch have not been reported in the literature.

1.3 OSMA Switches

Optical shared-memory architecture (OSMA) switches are defined here as all-optical packet switches that support

- completely shared buffering — each buffer cell can be used by any arriving packet;

- random access at the packet level to and from each buffer cell — each buffer cell can be read or written at each timeslot regardless of its "location" or the contents of the buffer cells;
- thru capability — packets are sent to the buffer only in the event of contention;
- fair dropping of packets — packets that cannot be buffered are selected for dropping without address favoritism;
- work conserving operation — output links will always be utilized to the fullest extent possible (i.e., there is no such thing as an idle output link when there are packets anywhere in the buffer or inputs that are destined to that output).

Physically, OSMA switches can be implemented in different ways. Perhaps the most obvious is the all-optical Starlite switch shown in Figure 2. The all-optical Starlite switch consists of an optical cross-connect and one or more optical buffer loops, each capable of storing a single optical packet. Control information such as packet addresses may be provided by low bandwidth out-of-band signaling techniques [20]. The only electronic controls needed are those used to extract the control information and to operate the cross-connect. Unlike the electronic Starlite switch, the switch control is required to drop packets fairly. Because the all-optical switch fabric is controlled by a single controller, this is not a difficult requirement to meet. Another way

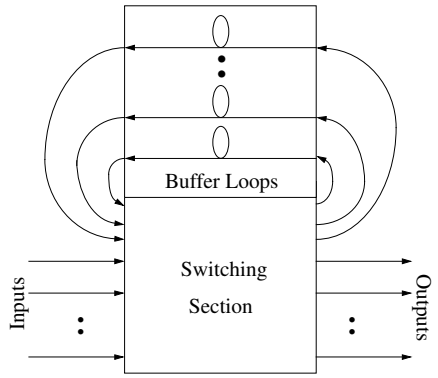


Figure 2: The all-optical Starlite OSMA switch.

to build an OSMA switch is to use a single-fiber loop, but allow this loop to hold multiple packets, each at a different wavelength. This implementation is known as the multiwavelength fiber loop memory switch (MFLMS) [21], [22], also known as the fiber-loop buffer memory (FLBM) switch [12], [13], which is shown in Figure 3. Finally, there are also exotic optical buffering technologies such as "light freezing" [23] which are promising, but currently far from practical.

The OSMA logical operation cycle is as follows:

1. Packets arrive at the inputs. It is assumed that there is some method to synchronize arrivals at the packet level. The destination addresses of the incoming packets are extracted.

2. As many packets as possible are routed out of the buffer. If there is contention, packets will be randomly chosen for routing from the contenders.

3. As many packets as possible are routed from the inputs to the outputs. In case of contention, packets are randomly chosen for routing.

4. If some input packets could not be routed out, they are stored in the buffer. If the buffer cannot hold all of these packets, a maximal subset is randomly chosen for buffering and the rest are dropped.

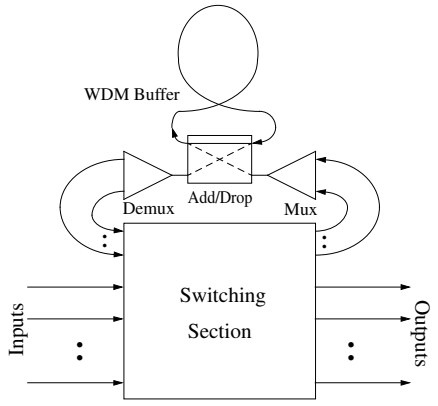


Figure 3: The multiwavelength fiber loop memory OSMA switch.

Of course, there may be variations, such as choosing from contending packets based on priority, buffer age, or sequence number. However, like the ESMP switch, these will not affect the loss rate or the average number of packets in the buffer.

The OSMA switch has a number of advantages. Unlike the ESMP switch, the OSMA switch does not have to process *any* of the packet data other than the control information. As a result, the data rate and format can be altered without requiring changes to the switch (i.e., data transparency). Furthermore, there is no need for modulated output lasers. Because packet buffering is done only in the event of contention, the loss and delay performance is better than that of an ESMP switch with the same size buffer (Figure 4). It should also be noted that increasing the size of the packets is relatively easy to accommodate by lengthening the fiber loops and/or increasing the data rate. This contrasts with the ESMP switches, which require increases in buffer memory to match the larger packet size.

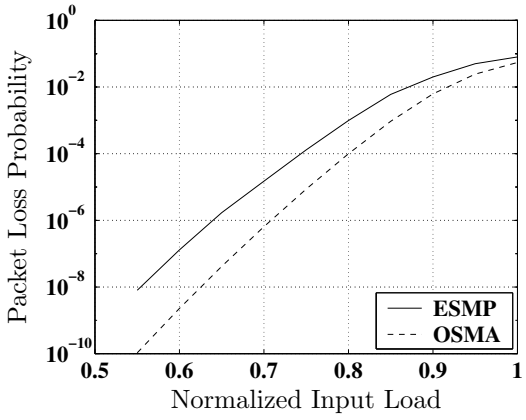


Figure 4: OSMA versus ESMP loss performance for 4×4 switches with 16 buffer cells under random traffic. ESMP data taken from Pattavina and Gianatti's exact vectorial model [24, page 406].

However, there are a number of disadvantages. If there is no regeneration and retiming of the packets, noise, timing skew, cross-talk and optical signal attenuation will accumulate as the number of switch hops increases. For the same reasons, there

may be a maximum length of time a packet can remain in a buffer loop until it becomes unacceptably degraded. There are promising all-optical means to perform packet regeneration and retiming. However, such technology is not yet practical [25], [26]. Therefore, making an optical buffer that is able to preserve packets for the required amount of time is difficult and expensive [27]–[30].

Furthermore, there are severe limitations on the size of all-optical cross connects for high speed packet switching. Currently, a 8×8 all-optical switch fabric is considered to be quite large [7]. Most experimental demonstrations of all-optical packet switches utilize switch fabrics with less connectivity, such as 2×2 or 4×4 [4], [29], [31]–[34]. Therefore, the cost per buffer cell is several orders of magnitude higher than the RAM used in the ESMP switch. Consequently, current OSMA buffers must be far smaller, in terms of the number of packets held, than their ESMP counterparts.

1.4 Non-OSMA Switches

There are all-optical shared-memory switches that are not members of the OSMA family. In fact, physical implementation difficulties may favor a non-OSMA design. A few of the more important non-OSMA switches will now be briefly mentioned.

1.4.1 Karol's Shared-Memory Optical Packet (SMOP) Switch

Mark J. Karol proposed the shared-memory optical packet (SMOP) switch shown in Figure 5 [35]. The design is similar to the all-optical Starlite switch (Figure 2), but with recirculation delay lines of possibly different length. In the most general form, the delay lines can be of arbitrary length, the only constraint being that each can store an integer number of packets. The all-optical Starlite switch is a special case of Karol's SMOP switch in which all the delay lines are one packet long. However, in the typical configuration suggested by Karol, each additional delay line is one packet longer than its predecessor. Thus, such a switch with i delay lines can buffer up to $\frac{(i)(i+1)}{2}$ packets. The inspiration for this design derives from the idea that packets that need buffering

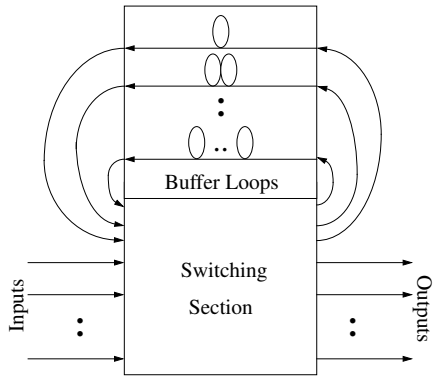


Figure 5: Karol's shared-memory optical packet (SMOP) switch.

should be "rescheduled" so that they do not (re)contend in future timeslots. This strategy can pay off when the number of delay lines (and resulting switch fabric size), rather than the number of buffer cells per se, is the constraining factor. Karol's typical configuration generally out performs an all-optical Starlite OSMA switch with the same number of delay lines.

However, using delay lines with lengths greater than one packet timeslot introduces a number of disadvantages. Because packets can be written/read only at the start/end of each delay line, true random access to each buffer cell is not possible. Thus, switching decisions cannot be globally (re)optimized at each timeslot. Buffered packets cannot preempt arriving packets based on priority without compromising performance. Furthermore, packet delay through the switch (latency) increases with the lengths of the delay lines.

For these reasons, when comparing switches with the same total number of buffer cells, OSMA switches offer superior performance over designs that use delay lines longer than one packet timeslot. Therefore, the OSMA design is typically to be preferred when the number of buffer cells is the constraining factor rather than the number of delay lines (and switch fabric size). For instance, the MWFLM switch implementation (Figure 3) favors the OSMA approach.

1.4.2 Haas' Staggering Switch

Zygmunt Haas proposed the "Staggering switch" (Figure 6) [36]–[41]. The Staggering switch uses delay lines with each additional delay line having one additional packet delay more than its predecessor, as with the typical configuration of Karol's SMOP switch. However, Haas' design is purely "feedfoward" and has a major advantage of not requiring the use of recirculation loops. Karol's SMOP switch can be viewed as being a Staggering switch in a feedback configuration. Depending on the various parameters and constraints, either the feedfoward or feedback configuration may be superior [40]. In particular, the feedforward configuration can outperform its feedback counterpart if the recirculation loops of the latter cannot maintain packet integrity for more than a few circulations. The Staggering switch has the same basic disadvantages as Karol's SMOP switch and cannot compete with the performance of an OSMA switch with the same number of buffer cells.

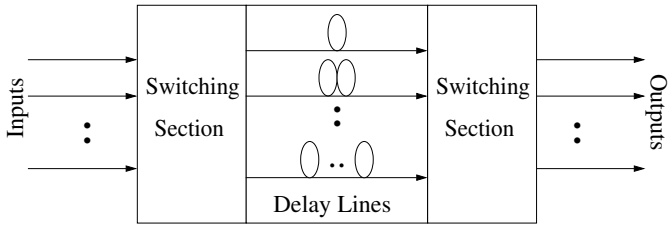


Figure 6: Haas' Staggering switch.

1.5 Channel Grouping

It is desirable to reduce contention as much as possible. This is especially true in all-optical networks because of the expensive nature of optical buffer loops.

Channel grouping has been proposed as a means to reduce or eliminate contention [10], [11], [42]–[46]. With channel grouping, each output (and/or input) port

of a switch is expanded via parallel channels so that a given link[1] can carry more than one packet at a time (Figure 7). Physically, this may be accomplished via parallel

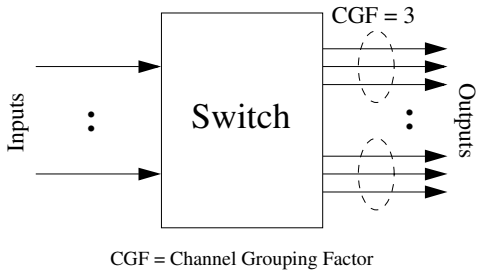


Figure 7: A switch with channel grouping on the outputs.

fibers or WDM within each fiber. In effect, the multichannel interswitch links serve as external "buffers" which ease the demands placed on the internal buffer of the switch. The number of parallel channels in a link is called the channel grouping factor (CGF). A CGF of one implies that there is no channel grouping.

Packets that can take advantage of these parallel pathways do not suffer the amount of degradation that would occur if they had been held for multiple circulations in a buffer loop. Furthermore, the packet loss rate and needed buffer size can be reduced to arbitrarily low values by increasing the CGF on the outputs.

However, there are drawbacks to channel grouping. Obviously, the additional pathways add to switch complexity and size. For all-optical WDM systems, this will likely require the use of all-optical wavelength converters. Perhaps a less obvious and more serious problem is that these additional channels will have to be connected to another switch or an end user (data sink). This next stage will then have to be able to process a higher peak bandwidth because of the multiple packets arriving simultaneously in each link. Furthermore, the network will utilize the available bandwidth less

[1]In this work, "link" is synonymous with "channel group."

efficiently if these additional channels are used only when contention occurs (total switched bandwidth remains the same).

Channel grouping should be evaluated against other alternatives such as increasing the capacity of each link via a speed-up so as to lower the normalized link load to reduce the probability of contention. The cost of implementing each alternative may not be proportional to the corresponding change in capacity because of the presence of factors such as fixed costs (e.g., the initial cost to provide the first channel) and nonlinearities in the cost/benefit curve (e.g., decreasing returns-to-scale).

1.6 Banyan Networks

Because of the limitations on the maximum size of a single switch and the need for geographically distributed routing, switching elements[2] (SE) are usually interconnected to form a network. Banyan networks are particularly promising for all-optical use because of their "self-routing" property. With self-routing, each switch can make its own routing decisions, based solely on the packets arriving to it, without the need to consider the routing decisions or status of the other switches in the network. Self-routing promotes high-speed operation by allowing the parallel processing of routing decisions by all the switches. This is especially important in all-optical networks because the packets have a time-of-flight nature. On a related note, this work will assume that there are no interstage flow control mechanisms (back pressure, etc.). The Banyan family encompasses a number of networks of different topologies such as the Baseline, N-cube, and Omega [17, chapter 2], [47].

An $N \times N$ Banyan network can be constructed by using $\log_n N$ stages of $n \times n$ switches, with each stage having N/n switches. Interstage connections are performed according to permutations such as butterfly, shuffle, identity, and their inverses, depending upon which particular member of the Banyan family is desired. As is typically

[2] A switch within a network is sometimes referred to as a "switching element" to emphasize its modular role.

done, this work will confine itself to cases in which n and N are powers of two. An example of a Banyan network is shown in Figure 8.

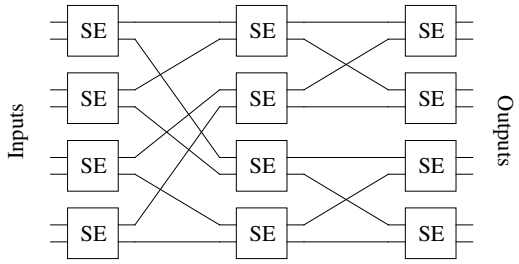


Figure 8: An 8×8 Baseline Banyan network using 2×2 switches.

Banyan networks lend themselves well to analysis, in part because the input ports of each switch do not have paths to a common upstream switch. Hence, there are no correlations among the arriving packets at the different inputs of each switch. If one further assumes that each switch self-routes and that packet destination addresses are uniformly and randomly assigned, a given switch can be analyzed as though it were the end point of a network consisting of a "cascaded" series of independent switches.

1.7 Markov Models

The use of Markov models to analyze switch performance will now be briefly reviewed. In a Markov model, all of the historical dependencies of the system to be analyzed are contained in its current "state." Many physical systems can be modeled as Markovian. Markov processes in which time is discrete are referred to as "chains."[3]

[3]Some authors use this term to refer to Markov processes with discrete *states*. In this work both time and the finite state spaces are discrete. Therefore, there is no ambiguity here.

1.7.1 The Use of Markov Chains to Analyze Systems

Once a suitable state space $\boldsymbol{\mathcal{S}}$ is determined for the system, the transition probability matrix $\mathbf{P}$ must be generated. The elements of $\mathbf{P}$, $P_{x,f}$, are the probability of transitioning from state number x to state number f. The total number of states is represented[4] by $|\boldsymbol{\mathcal{S}}|$. Therefore, $0 \leq x, f < |\boldsymbol{\mathcal{S}}|$. The difficulty in generating $\mathbf{P}$ usually determines whether the model is computationally tractable. Because $\mathbf{P}$ has $|\boldsymbol{\mathcal{S}}|^2$ elements, $\mathbf{P}$ will rapidly consume computational resources, such as memory, with increasing state space. Furthermore, the number of calculations required to fill $\mathbf{P}$ can grow quite large, depending not only on the size of $\mathbf{P}$, but also on the probabilistic complexity of the individual state transitions.

It is a property of Markov chains that if the probability distribution of the states at time step t is given by $\boldsymbol{\pi}^t$, where each element π_i^t is the probability of the system being in state number i at step t, the probability distribution of the states at the next time step can be found using the following relation:

$$\boldsymbol{\pi}^{t+1} = \boldsymbol{\pi}^t \mathbf{P}, \quad \text{where } t = 0, 1, 2, \ldots \tag{1}$$

Iterating this recurrence relation, we can write

$$\boldsymbol{\pi}^t = \boldsymbol{\pi}^0 \mathbf{P}^t, \tag{2}$$

where $\boldsymbol{\pi}^0$ is any given valid initial state distribution. The limit

$$\boldsymbol{\pi} = \lim_{t \to \infty} \boldsymbol{\pi}^t = \lim_{t \to \infty} \boldsymbol{\pi}^0 \mathbf{P}^t, \tag{3}$$

is known as the stationary, or steady-state, probability distribution. Physically, the steady-state probabilities are the probabilities of finding the system in each state as it is observed in operation for an infinite amount of time. If certain properties (discussed in the next section) hold for the Markov chain, the limit in Equation 3 is guaranteed

[4]Terminology: The cardinality operator, | |, when used with a set, denotes the number of items in the set. With a matrix or vector, it denotes the number of elements in the matrix or vector.

to exist and will converge to a unique $\boldsymbol{\pi}$, irrespective of the particular choice of $\boldsymbol{\pi}^0$, i.e., the Markov chain loses memory of its initial starting point.

Once the steady-state probabilities are known, the performance of the entire system can be calculated. Typically, the desired performance metric (loss, buffer occupancy, etc.) is calculated for each state and then the steady-state probabilities are used to extend the results to the overall system. For each state, the performance metric could be explicitly "hard-coded" into the state name or could be inferred analytically from the properties of the given state.[5]

1.7.2 Existence of the Steady-State Probabilities

If a Markov chain is homogeneous, irreducible, and has one ergodic state set, it is guaranteed that the steady-state probabilities exist, and are unique, and that Equation 3 can be used to obtain them [48], [49]. A Markov chain is homogeneous if the transition probabilities are not a function of time. A Markov chain is said to be irreducible if every state is reachable from every other state, albeit possibly through intermediate states. Irreducibility generally precludes the presence of states, or groups of states, that are "absorbing." A state is ergodic if it is positive recurrent and aperiodic. Positive recurrent implies that the expected number of steps in a cycle beginning and ending in the state is finite. (i.e., it will not take an infinite length of time for a state to return to itself.) A state is said to be periodic if it can return to itself only after some multiple of the integral period γ, $\gamma > 1$. All states in a periodic Markov chain have the same period γ. Aperiodic is synonymous with nonperiodic.

All chains in this work are homogeneous. The irreducibility condition is generally satisfied for models in this work as the states are based on the contents of the switch buffers and traffic sources, and every state is reachable from every other state if the ground state (empty buffer and all input sources idle) is used as an intermediary.

[5]The difference between the two is somewhat analogous to the approaches used in the Moore and Mealy hardware state machines, respectively.

Additionally, because the number of states is finite, it follows that the states must be positive recurrent. Finally, the ground state can generally reach itself within one time slot (no packets arrive and the sources remain off), which implies that it and, consequently, all the states are aperiodic.

Chapter 2

Prior Work

Prior work with shared-memory switches and/or channel grouping will now be briefly reviewed.

2.1 Karol and Hluchyj

Some of the earliest work with electronic shared-memory switches was done by Mark J. Karol and Michael G. Hluchyj [50], [51]. In these works, the superiority of shared queuing over input or output buffering was shown. Karol's shared-queue switch was of the Starlite architecture. The models were not exact as groups of packets in the buffer with the same destination addresses were assumed to be independent of each other. This assumption becomes increasingly true as the number of switch output ports increases. However, for switches with less than about 16 output ports, the results are quite inaccurate. The input traffic was uniform and random.

Karol and Hluchyj also did some work with switches that utilize "input smoothing" which is functionally like channel grouping. However, their analysis of input smoothing was confined to bufferless switches [50], [51].

Finally, Karol did some work with all-optical packet switches [35], [52]. Karol's shared-memory optical packet switch was discussed in Section 1.4.1. Analysis was confined to simulations of SMOP switches under uniform random traffic.

2.2 Szymanski and Shaikh

Ted Szymanski and Salman Shaikh did some of the first work with channel-grouped (dilated) Banyan networks [42]. Their approximate models were for networks of input and output buffered switches under uniform random traffic. Because the buffer states of switches in different stages were treated as being independent, these models were

not very accurate. However, Szymanski and Shaikh's work was among the first to show the potential benefits of using channel grouping as a means to reduce packet loss probabilities and/or increase throughput.

2.3 Liew and Lu

Soung Liew and Kevin Lu analyzed asymmetric packet switches that use channel grouping [43], [44], [53]. Input and output buffered channel-grouped switches under random and bursty traffic were studied. Only the maximum throughput was calculated as the switch buffers were assumed to be always saturated. A further assumption was that the packets in different channels of a link were independent from each other. There was no consideration of shared buffering. In [53], a three-stage structure (network) for constructing very large switch fabrics out of much smaller switches was introduced.

2.4 Lin and Silvester

Arthur Lin and John Silvester developed an exact model of an output-buffered switch using channel grouping on the output links under random and deterministic traffic [54]. Networks and/or bursty traffic were not considered.

2.5 Izmailov and Haas

Rauf Izmailov and Zygmunt Haas developed an analytic model of Haas' Staggering switch (the non-OSMA switch discussed in Section 1.4.2) under random traffic [41]. Exact analysis was possible only for switches with up to three delay lines after which an upper/lower approximate analysis was used.

Haas also evaluated via simulation the performance of the Staggering switch under bursty traffic as well as within a network [38]–[40]. Although Haas proposed the use of the Staggering switch in multiwavelength systems [38], [39], the independent use of multiple wavelength channels within each switch (channel grouping) to resolve contention was not considered.

2.6 Turner and Bianchi

Giuseppe Bianchi and Jonathan S. Turner analyzed delta networks constructed of ESMP switches [55], [56]. A variety of approximate models were used. The uniform scalar model used a single value, which indicates the total number of packets in the buffer, to represent the switch state. The bidimensional model expanded the state space of the uniform scalar model to include the number of active switch outputs. A third model, the interval and threshold method, attempts to reduce the state space of the bidimensional model by grouping the numbers of active switch outputs into intervals (state grouping). The network traffic was uniform and random.

Although computationally tractable, these models are quite inaccurate, even for a single switch. For networks, the inaccuracy is far worse because interstage state correlation was not considered. The accuracy does improve somewhat with increasing switch and decreasing buffer sizes.

2.7 Pattavina, Monterosso and Gianatti

Achille Pattavina, along with Alberto Monterosso and Stefano Gianatti, did some of the most extensive work with ESMP switches and Banyan networks constructed from them [17], [24], [57], [58]. A number of different models were presented, such as the vectorial, scalar, bidimensional, tridimensional, and four dimensional. Both random and bursty traffic cases were analyzed.

For an isolated switch, the vectorial model is exact. However, it was never extended to handle bursty traffic — presumably because of the large number of states that would be required. Consequently, the vectorial model is inaccurate for networks because it makes unrealistic assumptions about the independence of the buffer states of switches in different stages. The scalar model is like that of Karol's work and is extremely inaccurate. The n-dimensional models are named after the n-tuples used to describe the state of each switch. The bidimensional model is similar to Turner's bidimensional model, except that the two state terms describe the number of packets

in the buffer destined to a tagged and nontagged switch outlet(s), rather than the total number of buffer packets and number of active outputs, as is done with Turner's approach. The tri- and four-dimensional models build on the bidimensional model by adding terms that describe the states of bursty input sources. Although the tri- and four-dimensional models greatly improve on the accuracy of the bidimensional model when used on a multistage network, they are still approximations. As such, they can generate results that are off by an order of magnitude or more — even for a single switch. Only the results for the overall network were presented. There was no analysis of how switch performance can vary between the stages of the network. Channel grouping was not considered.

2.8 Montagna, Paglino and Meyer

Sergio Montagna, Roberto Paglino and John Meyer developed an exact and an approximate model for ESMP switches under random and bursty traffic [59]. Their approximate model was somewhat unique in that it could internally exploit the use of their exact model (on smaller component "subsystems") to reduce the required computation time and improve the accuracy of the results for switches too large to be directly analyzed by the exact model. Their models are not applicable to channel-grouped switches or to networks.

2.9 Fong and Singh

Simon Fong and Samar Singh developed several approximate models for ESMP switches under bursty traffic [60]–[62]. These models are somewhat similar to those of Pattavina and Turner, but with variations and improvements such as the consideration of the "hot-spot pushout" buffer management protocol. Their models are typically quite accurate for loss rates, but less so for other performance metrics (e.g., buffer occupancies and packet delays).

Fong and Singh also did some work with networks of ESMP switches [63]. Several different backpressure flow control schemes were compared. The accuracy of the results was surprisingly good despite the use of a relatively simple buffer state description. The model will over or under estimate loss rates depending on the network parameters. Channel grouping was not considered.

2.10 Saleh and Atiquzzaman

Mahmoud Saleh and Mohammed Atiquzzaman did some of the more recent work with networks of ESMP switches [64]–[66]. Their work focuses mainly on variations of Pattavina's and Turner's work, such as analyzing global flow control protocols or more general forms of unbalanced traffic. Saleh and Atiquzzaman's models are not applicable to switches using channel grouping or to those under bursty input traffic.

2.11 Danielsen

Soeren Danielsen along with Benny Mikkelsen, Carsten Joergensen, Terji Durhuus and Kristian E. Stubkjaer analyzed multiwavelength [45] (channel-grouped) switches of three different configurations, each of which used feedforward delay lines similar to Haas' Staggering switch. The channel traffic was independent, uniform and random. Exact results could be obtained for packet loss performance. The ability to wavelength convert packets to avoid contention was shown to greatly improve switch performance. Bursty traffic or networks were not considered.

2.12 Bergstrom

The most extensive work with OSMA switches to date has been by Peter D. Bergstrom Jr. [10], [11]. The Bergstrom reduced Markov chain (RMC) model provided an exact analysis of symmetric channel-grouped OSMA switches under uniform, random traffic. Unfortunately, the RMC state description included the throughput, which resulted in the rapid growth of the state space with increasing switch size and made description

of the state space difficult. This large state space precluded the possibility of investigating switches under bursty traffic. Furthermore, the calculation of the packet loss probability from throughput presents numerical difficulties for loss rates less than 10^{-15} because of the typical floating point accuracy limits of most computational platforms. Networks of switches were not considered.

2.13 Chia and Hunter

M. C. Chia and David K. Hunter *et al.* developed an approximate model for an all-optical non-OSMA switch related to Haas' staggering switch under uniform random traffic [46]. Chia and Hunter's designs (both feedfoward and feedback designs were proposed) utilized multiple blocks of delay lines with delay line selection within each block being controlled via wavelength conversion. Multichannel link operation was proposed, but this required the switch structure to be independently duplicated for each channel. Thus, delay lines could not be globally shared by traffic of different input wavelengths. Their model was very accurate in predicting packet loss, but less so for other switch performance metrics (e.g., packet delay). Bursty traffic or networks were not considered.

2.14 Singh, Kushwaha and Bose

Yatindra Singh, Amit Kushwaha and Sanjay Bose did some very recent work with multiwavelength fiber-loop buffer memory (FLBM) switches under random traffic [12], [13]. Both exact and approximate models were developed. These models had some unique features including the ability to take into consideration some practical limitations of the optical buffer including read/write access restrictions and finite packet lifetime. The buffer states of the exact model did not employ buffer state reduction techniques such as those used in the models of Bergstrom, Montagna, Gianatti and Pattavina. Consequently, the exact model's state space scales especially poorly with increasing switch size. Switches larger than 4×4 were not analyzed. Furthermore, to

simplify analysis, the models require the switch to drop packets unfairly, in a deterministic fashion which favors the dropping of packets with lower addresses. Bursty traffic, networks or the use of channel grouping were not considered.

Chapter 3

OSMA Switches Under Random Traffic

The exact Markov model of the OSMA switch under random, uniform traffic will now be developed. The switch parameters are as shown[1] in Figure 9, where m is the number of buffer cells, h is the number of input channel groups, g is the number of output channel groups, and s and r are the channel grouping factors of the inputs and outputs, respectively. For the case of symmetry in the number of input and output channel groups, $h = g = n$. For symmetric channel grouping factors, $s = r = c$.

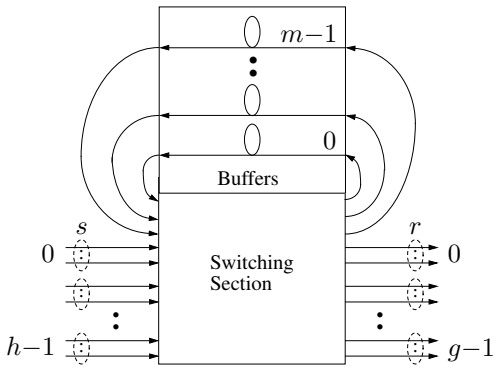


Figure 9: OSMA switch parameters (all-optical Starlite version shown).

3.1 The Uniform Random Source (RS)

The Markov model for a uniform, random source is shown in Figure 10.

The source consists of a single state that is completely described by the parameter p, which is the probability of emitting a packet in each timeslot. These sources are also referred to as "Bernoulli" or "geometric" sources because the presence or

[1]The all-optical Starlite implementation is used here because it is easy to clearly illustrate. Of course, the parameters mean the same thing in other OSMA implementations.

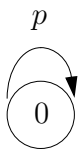


Figure 10: The random source.

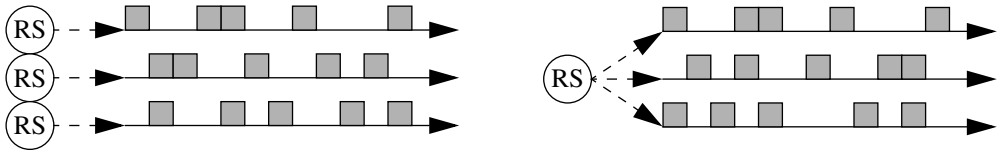
absence of packets in each timeslot is a Bernoulli random variable and the packet interarrival times have a geometric distribution. Because the source has only one state, it is completely memoryless. The average normalized load presented by the source, σ, is trivially determined by p, ($\sigma = p$).

For this work, "uniform" refers to the fact that each packet's destination address is randomly and uniformly selected from all the possible destinations.

3.1.1 The Extended Random Source (ERS)

Interfacing a random source to drive a multichannel link can be done in two obvious ways (see Figure 11). Independent sources can drive each channel (Figure 11(a)) or a single source can be sped-up to drive the entire link (Figure 11(b)). Because the random source is memoryless, these two approaches yield identical traffic. In both cases, the probability, q_i, that i ($0 \leq i \leq c$) packets are in the link in any given timeslot has a binomial distribution:

$$q_i = \binom{c}{i} p^i (1-p)^{c-i}, \tag{4}$$



(a) Drive independently (b) Speed-up a single source

Figure 11: Interfacing a random source to a multichannel link.

where c is the channel grouping factor of the link.[2] The average link load presented by either the sped-up or independent random source(s) is unaffected by c, ($\sigma = p$).

A more general memoryless random source, the extended random source (ERS), can be had by allowing an arbitrary distribution for the number of packets in the link (Figure 12), where p_i is the probability of emitting i packets into the link in any given timeslot and $\sum_{i=0}^{c} p_i = 1$. By setting p_i to conform to the binomial distribution, ($p_i = q_i$ of Equation 4), the ERS traffic will be equivalent to that of driving the channels independently.

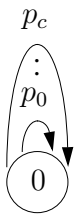


Figure 12: The extended random source

The normalized link load presented by an ERS source is given by

$$\sigma = \frac{\sum_{i=1}^{c} i p_i}{c}. \tag{5}$$

3.2 State Descriptions

For a given Markov system state space, $\boldsymbol{\mathcal{S}}$, $\boldsymbol{\mathcal{S}}_x$ denotes state number x and $|\boldsymbol{\mathcal{S}}|$ is the total number of states (cardinality of $\boldsymbol{\mathcal{S}}$) of the system to be analyzed. Therefore, $0 \leq x < |\boldsymbol{\mathcal{S}}|$. $\mathcal{S}_{x,i}$ is the value of element i in the tuple in the "name" of state x.

Techniques for describing the states of an OSMA switch in exact models will now be formulated. For uniform random traffic, the state of the system composed of

[2]This equation can be extended to handle $p = 1$ by defining $0^0 \equiv 1$. Physically, this means that the link will always be 100% utilized — anything less is impossible, i.e., $q_i = 1$ for $i = c$, $q_i = 0$ otherwise.

the switch is completely determined by the state of the buffer. Hence, $\boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{B}}$, where $\boldsymbol{\mathcal{B}}$ is the set of all buffer states. Although $\boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{B}}$ under random traffic, this is not necessarily true for other traffic models. Throughout this chapter, both $\boldsymbol{\mathcal{B}}$ and $\boldsymbol{\mathcal{S}}$ will be used such that the developed equations will have greater generality. It should also be noted that a bufferless switch is a special case that has the single state of an empty buffer.

3.2.1 General Buffer Description

One of the most general models can be had by using the destination addresses in each distinct buffer cell to represent the state. A state $\boldsymbol{\mathcal{B}}_x$ is then an m-tuple of values 0 to g, where g, an out of range address, denotes an empty buffer cell.[3] Hence, $0 \leq \mathcal{B}_{x,i} \leq g$, $0 \leq i < m$. $|\boldsymbol{\mathcal{B}}|$ is given by

$$|\boldsymbol{\mathcal{B}}| = (g+1)^m. \tag{6}$$

Combinatorially, $|\boldsymbol{\mathcal{B}}|$ represents the number of ways to put m distinguishable balls into $g+1$ distinguishable urns.[4] The exponential growth of $|\boldsymbol{\mathcal{B}}|$ with increasing buffer size makes this approach impractical for all but the most trivial of switches. However, the states do allow analysis of the occupancy statistics of the various buffer cells as well as statistics of the switching configurations of the cross-connect.

3.2.2 Buffer Cells are Indistinct

It is usually unimportant which particular buffer cell a packet occupies. A major reduction in the state space can be achieved by allowing the buffer cells to be indistinct from each other. A state $\boldsymbol{\mathcal{B}}_x$ is then described by a $(g+1)$-tuple with $\sum_{i=0}^{g} \mathcal{B}_{x,i} = m$. The $\mathcal{B}_{x,i}$ is the number of packets in the buffer destined for channel group i, with channel group $i = g$ indicating the free buffer cells. Combinatorially, $|\boldsymbol{\mathcal{B}}|$ represents

[3]The output channel groups have addresses 0 to $g-1$. This convention allows a given channel group address i to directly correspond to an index within a state tuple $\mathcal{B}_{x,i}$.

[4]This corresponds to m distinguishable buffer cells each of which holds one of $g+1$ possible values.

the number of vectors of length $g+1$ whose elemental sum is m. An alternative viewpoint of $|\boldsymbol{\mathcal{B}}|$ is that it is the number of ways to put m indistinguishable balls into $g+1$ distinguishable urns (the number of ways to select m objects from $g+1$ with replacement). Therefore, $|\boldsymbol{\mathcal{B}}|$ is given by

$$|\boldsymbol{\mathcal{B}}| = \left\langle \begin{matrix} g+1 \\ m \end{matrix} \right\rangle = \binom{g+m}{m} = \frac{(g+m)!}{m!g!}. \tag{7}$$

Unfortunately, $|\boldsymbol{\mathcal{B}}|$ is still too large for most OSMA switch sizes of interest. However, this state description does have the ability to analyze switches under nonuniform traffic in which some output destinations are favored over others.

It should be noted that because $\mathcal{B}_{x,g}$, the number of free buffer cells for $\boldsymbol{\mathcal{B}}_x$, is completely determined by m and $\sum_{i=0}^{g-1} \mathcal{B}_{x,i}$, it can be omitted when writing the state name. This shorthand form of the state name is a g-tuple consisting of the number of packets destined to each output link.

3.2.3 Destination Addresses are Indistinct

If the input traffic is uniform and random with respect to the destination addresses of the incoming packets, a further reduction in the state space can be achieved. Under these conditions, the *absolute* addresses of the packets in the buffer are of no importance. It is the relative numbers of packets addressed to each destination that determines the properties of interest for each state. For example, for a switch with six or more buffer cells and four output channel groups, the state [4,1,1,0], in which four, one, and one buffer packets are addressed to channel groups 0, 1, and 2, respectively, is considered to be equivalent to buffer states [0,1,4,1], [1,1,0,4], etc. Hence, the ordered state represents all the other permutations (i.e., state lumping). In this manner, a rather large reduction in the required state space is obtained. Combinatorially, the states are the ways in which b can be partitioned into parts not exceeding g in length, where b is the number of packets in the buffer (buffer occupancy), $0 \leq b \leq m$. An

alternative interpretation is that the states are the ways b indistinguishable balls can be placed into g indistinguishable urns.

Let the operator $\boldsymbol{\Psi}(k)$ generate the set of unrestricted partitions (no limits on the length of the partition or on the values of the elements) of the nonnegative integer k. Likewise, $\boldsymbol{\Psi}(k,\omega)$ is the set of partitions of k not exceeding ω in length and $\boldsymbol{\Psi}(k,\omega,\theta)$ has the further restriction that each element not exceed θ, $(\boldsymbol{\Psi}(k) = \boldsymbol{\Psi}(k,k) = \boldsymbol{\Psi}(k,k,k))$. A partition of k may have a length less than ω. In such cases, $\boldsymbol{\Psi}(k,\omega)$ will be defined such that zeros are appended, if needed, to ensure that the returned partitions always have ω elements. This is done purely for computational convenience, and the appended zeros in no way alter the partition "name." Furthermore, $\boldsymbol{\Psi}(0,\omega)$ is defined in this work as consisting entirely of a single vector of ω zeros. $\boldsymbol{\Psi}_i(k)$ is the ith partition and $\Psi_{i,j}(k)$ is the jth element in the ith partition $(0 \leq i < |\boldsymbol{\Psi}(k)|,\ 0 \leq j < \omega = k)$.

Therefore, $\boldsymbol{\mathcal{B}}$ is given by

$$\boldsymbol{\mathcal{B}} = \bigcup_{b=0}^{m} \boldsymbol{\Psi}(b,g), \tag{8}$$

where the set union operation is performed over every possible buffer occupancy value b. $|\boldsymbol{\mathcal{B}}|$ is given by

$$|\boldsymbol{\mathcal{B}}| = \sum_{b=0}^{m} |\boldsymbol{\Psi}(b,g)|. \tag{9}$$

For a more detailed discussion of partitions of integers and the $\boldsymbol{\Psi}$ function used in this work, see Appendix A.

$|\boldsymbol{\mathcal{B}}|$ is quite reasonable for OSMA switch sizes of interest. The model developed in this work uses this state description. It is also the basis of Monterosso and Pattavina's "vectorial" model [24], [57]. Bergstrom's RMC model used a similar state description, but it had an added throughput term in each state [10], [11]. The OSMA model of this work will be referred to as the "partition" model because of the origin of the state names.

3.3 Calculation of the Transition Probabilities

To generate the transition probability matrix $\mathbf{P}$, each state must be evaluated against the probability of each possible input stimulus to the switch and every possible way of dropping packets, if packets need to be dropped. In this manner, the probability of every possible transition out of each state, and consequently $\mathbf{P}$, is determined. The specific stimulus presented to the inputs of a switch in any given timeslot is called an arrival vector[5], $\boldsymbol{\mathcal{A}}_y$. The set of all arrival vectors is designated by $\boldsymbol{\mathcal{A}}$. The total number of possible arrival vectors is given by $|\boldsymbol{\mathcal{A}}|$. It should be noted that a given arrival vector $\boldsymbol{\mathcal{A}}_y$, when acting on a given buffer state $\boldsymbol{\mathcal{B}}_x$, can result in several possible buffer state transitions, if and only if the switch has to drop packets. Therefore, the total number of computations needed to form $\mathbf{P}$ is at least $|\boldsymbol{\mathcal{B}}||\boldsymbol{\mathcal{A}}|$. $|\boldsymbol{\mathcal{A}}|$ can become extremely large. If care is not taken to keep $|\boldsymbol{\mathcal{A}}|$ to a manageable size, it will not be possible to calculate $\mathbf{P}$ in a reasonable amount of time and the model will not be viable. This situation can be true even if $|\boldsymbol{\mathcal{B}}|$ is small. The reduction of $|\boldsymbol{\mathcal{A}}|$, in a manner that preserves the exactness of the model is known as arrival vector reduction (AVR) and will be discussed in Section 3.3.3.

An arrival vector can contain information on the number of arriving packets, their destination addresses, and the input links they arrive on. For the purposes of this work, which particular link supplies a group of packets is unimportant because the input links all carry traffic with the same properties. Therefore, an arrival vector can consist solely of information on the number of arriving packets and their destination addresses. The elements of $\boldsymbol{\mathcal{A}}_y$, $\mathcal{A}_{y,i}$, are the number of incoming packets with destination addresses i.

[5]Notation: x will be used to indicate a particular state, y will be used to index arrival vectors and z will indicate a way of dropping packets.

3.3.1 The OSMA Operation Cycle

The steps an OSMA switch takes when transitioning from one buffer state, $\boldsymbol{\mathcal{B}}_x$, to another, $\boldsymbol{\mathcal{B}}_f$, ($\boldsymbol{\mathcal{B}}_x, \boldsymbol{\mathcal{B}}_f \in \boldsymbol{\mathcal{B}}$) will now be detailed. As described in Section 3.2.3, $\boldsymbol{\mathcal{B}}_x$ consists of an ordered g-tuple, with each value $\mathcal{B}_{x,i}$ ($0 \leq i < g$) indicating the number of packets in the buffer addressed to a "tagged" output channel group i. It is important to emphasize that $\mathcal{B}_{x,i}$ is *not* necessarily the number of packets addressed to physical output channel group i. Other works bypass this issue by having the switch dynamically reorder the physical destination addresses of packets in the buffer at the end of each operation [10]. However, with this other approach, some readers may incorrectly infer that additional switch complexity must be introduced to make the analytic model feasible. The approach used in this work is to state outright that the absolute physical addressing information has been removed from $\boldsymbol{\mathcal{B}}$ by requiring that all states consist of ordered elements. $\mathcal{B}_{x,i}$ corresponds to a single, *unspecified*, physical address. Furthermore, $\mathcal{B}_{x,j}$ ($i \neq j$) corresponds to a different, single, unspecified physical address. Computations will be done with tagged addresses. Unless otherwise stated, "address" or "output channel group (link)" will refer to "tagged address" or "tagged output channel group (link)," respectively.

At the beginning of the switch's operational cycle, the switch will route all the packets it can from the buffer. Then, the number of packets in each output channel group, O_i, is given by

$$O_i = \begin{cases} \mathcal{B}_{x,i} & : \quad \mathcal{B}_{x,i} \leq r \\ r & : \quad \mathcal{B}_{x,i} > r \end{cases}. \tag{10}$$

The number of unused (free) output channels in each channel group, U_i, is given by

$$U_i = r - O_i. \tag{11}$$

The buffer state changes to $\boldsymbol{\mathcal{B}}'_x$ as a result of the routing out of the packets:

$$\mathcal{B}'_{x,i} = \mathcal{B}_{x,i} - O_i. \tag{12}$$

Note that because $\mathbf{O}$ is an ordered g-tuple, $\boldsymbol{\mathcal{B}}'_x$ is also ordered and $\boldsymbol{\mathcal{B}}'_x \in \boldsymbol{\mathcal{B}}$. If no additional packets arrive, $\boldsymbol{\mathcal{B}}'_x$ will be the state at the end of the cycle and $\boldsymbol{\mathcal{B}}_f = \boldsymbol{\mathcal{B}}'_x$. Otherwise, the switch then routes what it can from the incoming packets in the arrival vector $\boldsymbol{\mathcal{A}}_y$. Let α be the total number of arriving packets in $\boldsymbol{\mathcal{A}}_y$, $\alpha = \sum_{i=0}^{g-1} \mathcal{A}_{y,i}$. The packets that can be directly routed to each output channel group i are given by

$$V_i = \begin{cases} \mathcal{A}_{y,i} & : \quad \mathcal{A}_{y,i} \leq U_i \\ U_i & : \quad \mathcal{A}_{y,i} > U_i \end{cases}, \tag{13}$$

and $v = \sum_{i=0}^{g-1} V_i$ is the total number of packets directly routed. The total number of packets at each output channel group is now

$$O'_i = O_i + V_i. \tag{14}$$

Those packets that could not be routed, D_i, will have to be buffered and are given by

$$D_i = \mathcal{A}_{y,i} - V_i. \tag{15}$$

The total number of packets needing buffering, d, is given by

$$d = \sum_{i=0}^{g-1} D_i. \tag{16}$$

The total number of empty buffer cells after routing from the buffer, e, is given by

$$e = m - \sum_{i=0}^{g-1} \mathcal{B}'_{x,i}. \tag{17}$$

If $d \leq e$, the buffer can hold all the packets requiring buffering and there will be no packet loss. In this case, the buffer contents become

$$\mathcal{B}''_{x,i} = \mathcal{B}'_{x,i} + D_i \qquad (d \leq e). \tag{18}$$

However, D_i is not guaranteed to be ordered. $\mathcal{B}''_{x,i}$ must be sorted in decreasing order to be a valid state name[6] and a member of $\boldsymbol{\mathcal{B}}$. Hence, the final state of the buffer

[6] Again, the physical addresses are not changing here. The tagged addresses are being ordered so that $\boldsymbol{\mathcal{B}}_f$ matches an ordered state name that represents many permutations — one of which is $\boldsymbol{\mathcal{B}}''_x$. The switch does not actually perform this operation.

becomes

$$\boldsymbol{\mathcal{B}}_f = \text{ordered}(\boldsymbol{\mathcal{B}}''_x). \tag{19}$$

On the other hand, if $d > e$, there will have to be $l = d - e$ packets lost. $w = e$ packets will be randomly chosen for buffering from the d contending for the buffer. These packets will be referred to as the "winning" packets, W_i, ($\mathbf{W}$ is the "win vector"):

$$\sum_{i=0}^{g-1} W_i = w. \tag{20}$$

The losing packets will be designated $\mathbf{L}$ (loss vector) with

$$\sum_{i=0}^{g-1} L_i = l, \qquad W_i + L_i = D_i. \tag{21}$$

The buffer contents become

$$\mathcal{B}''_{x,i} = \mathcal{B}'_{x,i} + W_i. \tag{22}$$

However, W_i is not guaranteed to be ordered. $\mathcal{B}''_{x,i}$ must be sorted in decreasing order to be a valid final state name, $\boldsymbol{\mathcal{B}}_f$, as in Equation 19. The three equations above can be used even if no packets are to be dropped if one recognizes that in this event, $\mathbf{L}$ consists entirely of zeros and $\mathbf{W} = \mathbf{D}$.

Given that the switch is in state $\boldsymbol{\mathcal{B}}_x$ before the arrival vector, the probability of this particular buffer transition to $\boldsymbol{\mathcal{B}}_f$ occurring in the manner just described is given by $\mathcal{P}_{\mathcal{T}_\mathcal{B}}$

$$\mathcal{P}_{\mathcal{T}_\mathcal{B}} = \mathcal{P}_\mathcal{A}\mathcal{P}_L, \tag{23}$$

where $\mathcal{P}_\mathcal{A}$ is the probability of this arrival vector, $\boldsymbol{\mathcal{A}}_y$, occurring and $\mathcal{P}_L$ is the probability of choosing this particular way, $\mathbf{L}$, for the packets to be dropped. If no packets are to be dropped, $\mathcal{P}_L = 1$, as there is only one way not to drop any packets. Because

the switch buffer states *are* the Markov states ($\boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{B}}$), $\mathcal{P}_{\mathcal{T}_\mathcal{B}}$ is also the probability of transitioning between system states in the manner determined by $\boldsymbol{\mathcal{A}}_y$ and $\mathbf{L}$:

$$\mathcal{P}_\mathcal{T} = \mathcal{P}_{\mathcal{T}_\mathcal{B}}. \tag{24}$$

$\mathcal{P}_\mathcal{T}$ must be added[7] to the $P_{x,f}$ term in the transition probability matrix $\mathbf{P}$, resulting in the revised $P_{x,f}$, which is denoted here as $P'_{x,f}$. (This is not necessarily the *only* way to transition from $\boldsymbol{\mathcal{B}}_x$ to $\boldsymbol{\mathcal{B}}_f$ —there most likely will be other arrival vectors and/or loss vectors that can cause the same buffer state change.)

$$P'_{x,f} = P_{x,f} + \mathcal{P}_\mathcal{T}. \tag{25}$$

The probability of a given arrival vector $\boldsymbol{\mathcal{A}}_y$ occurring is determined by two independent characteristics of $\boldsymbol{\mathcal{A}}_y$ —the probability of the number of packets arriving, $\mathcal{P}_\alpha$, and the probability that the particular destination addresses will occur, $\mathcal{P}_\delta$. Hence, $\mathcal{P}_\mathcal{A}$ can be written as

$$\mathcal{P}_\mathcal{A} = \mathcal{P}_\alpha \mathcal{P}_\delta. \tag{26}$$

Substituting into Equation 23 yields

$$\mathcal{P}_{\mathcal{T}_\mathcal{B}} = \mathcal{P}_\alpha \mathcal{P}_\delta \mathcal{P}_L. \tag{27}$$

$\mathcal{P}_\alpha$ and $\mathcal{P}_\delta$ are dependent on the type of input sources used. They will be derived in the next sections.

$\mathcal{P}_L$ is unchanged throughout this work and will be developed now. The number of ways this $\mathbf{L}$ could have occurred is given by $\binom{D_0}{L_0}\binom{D_1}{L_1}\cdots\binom{D_{g-1}}{L_{g-1}}$. The total number of ways to drop l packets from d is given by $\binom{d}{l}$. Therefore, the probability of this particular $\mathbf{L}$ occurring is

$$\mathcal{P}_L = \frac{\binom{D_0}{L_0}\binom{D_1}{L_1}\cdots\binom{D_{g-1}}{L_{g-1}}}{\binom{d}{l}}. \tag{28}$$

[7]It is assumed that $\mathbf{P}$ is initialized to the zero matrix before the start of the algorithm.

3.3.2 Nonreduced Arrival Vectors

As mentioned previously, an arrival vector $\boldsymbol{\mathcal{A}}_y$ consists of elements $\mathcal{A}_{y,i}$, which are the number of incoming packets addressed to output channel group i. Each input channel can either be empty or have a packet addressed to one of g destinations. Because there are hs input channels, the total number of possible arrival vectors is given by

$$|\boldsymbol{\mathcal{A}}| = \left\langle \begin{matrix} g+1 \\ hs \end{matrix} \right\rangle = \frac{(g+hs)!}{g!hs!}. \tag{29}$$

Unfortunately, this can be a relatively large number for switches of interest and scales badly with increasing switch size. Nevertheless, it is instructive to derive $\mathcal{P}_\alpha$ and $\mathcal{P}_\delta$ for this $\boldsymbol{\mathcal{A}}$.

3.3.2.1 Probabilities of Nonreduced Arrival Vectors from RS

If each channel is driven independently within each input link by random sources (RS), there are hs independent channels supplying packets to the switch. The probability of the switch receiving α packets ($0 \leq \alpha \leq hs$) in any given timeslot is given by

$$\mathcal{P}_\alpha = \frac{(hs)!}{\alpha!(hs-\alpha)!}\, p^\alpha (1-p)^{hs-\alpha}. \tag{30}$$

This follows from Equation 4. The probability, $\mathcal{P}_\delta$, of these α packets having the addresses described by $\boldsymbol{\mathcal{A}}_y$ is given by

$$\mathcal{P}_\delta = \frac{\alpha!}{\mathcal{A}_{y,0}!\mathcal{A}_{y,1}!\cdots\mathcal{A}_{y,g-1}!} \left(\frac{1}{g}\right)^\alpha. \tag{31}$$

Each packet has a $1/g$ chance of being a particular address and the $\frac{\alpha!}{\mathcal{A}_{y,0}!\mathcal{A}_{y,1}!\cdots\mathcal{A}_{y,g-1}!}$ part of the equation enumerates all possible permutations of these addresses.

3.3.2.2 Probabilities of Nonreduced Arrival Vectors from ERS

If the input links are fed by ERS sources, Equation 30 no longer holds because the channels within a link are no longer independent of one another. Hence, the arrival

vector $\boldsymbol{\mathcal{A}}_y$ must be thought of as being delivered by h groups of inputs rather than by hs channels. $\mathcal{P}_\alpha$ is given by

$$\mathcal{P}_\alpha = \sum_{\boldsymbol{\Psi}(\alpha,h,s)} \left(\frac{h!}{K_0!K_1!\cdots K_s!}\, p_0^{K_0} p_1^{K_1} \cdots p_s^{K_s} \right), \tag{32}$$

where the summation is over all partitions $\boldsymbol{\Psi}(\alpha, h, s)$, K_i is the number of elements of value i in a given $\boldsymbol{\Psi}_j(\alpha, h, s)$ $(0 \leq j < |\boldsymbol{\Psi}(\alpha, h, s)|)$, and p_k is the probability of each ERS source emitting k packets. Equation 32 requires further explanation. The numbers of packets in each of the h input links, when ordered, form a partition in $\boldsymbol{\Psi}(\alpha, h, s)$. The probability of this ordered vector occurring is given by the $p_0^{K_0} p_1^{K_1} \cdots p_s^{K_s}$ part of Equation 32. The $\frac{h!}{K_0!K_1!\cdots K_s!}$ term accounts for all possible orderings (permutations). It is possible to rewrite Equation 32 so that the summation is over all combinations of choosing α from h groups, each with s members (no replacement), and do away with the $\frac{h!}{K_0!K_1!\cdots K_s!}$ term. However, this would greatly increase the computation time, as there would be many more values to sum.[8] Furthermore, which particular input link delivers which set of packets is unimportant with random traffic. It should also be noted that because α has a relatively small range, $(0 \leq \alpha \leq hs)$, $\mathcal{P}_\alpha$ can be precomputed for each α and the results stored in a look-up table, thus avoiding the need to recompute $\mathcal{P}_\alpha$ for each arrival vector $\boldsymbol{\mathcal{A}}_y$.

Because the address of each packet is independent of every other, even within the same link, Equation 31 remains valid for $\mathcal{P}_\delta$ under ERS traffic.

3.3.3 Arrival Vector Reduction (AVR)

Suppose $\mathcal{B}_{x,i} = \mathcal{B}_{x,j}$ for given valid i, j $(i \neq j)$. Furthermore, suppose the two corresponding elements within the arrival vector $\boldsymbol{\mathcal{A}}_y$ are swapped to result in a new arrival vector $\boldsymbol{\mathcal{A}}^\star_y$:

$$\mathcal{A}^\star_{y,i} = \mathcal{A}_{y,j}, \quad \mathcal{A}^\star_{y,j} = \mathcal{A}_{y,i}, \quad \mathcal{A}^\star_{y,k} = \mathcal{A}_{y,k} \quad \forall\ k \neq i, j. \tag{33}$$

[8] This assumes that the algorithm to generate partitions is not very much slower than that used to generate combinations. In practice, this is the case — it is usually faster to generate partitions.

$d, e, l, \mathcal{P}_{\mathcal{A}},$ and $\mathcal{P}_L$ are all unaffected by swaps and are equal to their $\star$ counterparts. Looking at the final states of the transitions with the switch dropping packets in the same way ($L_i^\star = L_j,\ L_j^\star = L_i,\ L_k^\star = L_k\ \ \forall\ k \neq i, j$), one finds that

$$\mathcal{B}''^{\star}_{x,i} = \mathcal{B}''_{x,j}, \quad \mathcal{B}''^{\star}_{x,j} = \mathcal{B}''_{x,i}, \quad \mathcal{B}''^{\star}_{x,k} = \mathcal{B}''_{x,k} \quad \forall\ k \neq i, j. \tag{34}$$

However, $\boldsymbol{\mathcal{B}}_f$ is an ordered $\boldsymbol{\mathcal{B}}''_x$. Therefore,

$$\boldsymbol{\mathcal{B}}^\star_f = \boldsymbol{\mathcal{B}}_f. \tag{35}$$

The conclusion is that permuting the elements in arrival vectors whose corresponding buffer state elements are all equal has no effect on the number of packets dropped by the switch or on the next state of the transition! This insight provides a way to greatly reduce the size of $|\boldsymbol{\mathcal{A}}|$, yet still retain an exact model.

The sets of identical elements within a buffer state are called buffer groups. The corresponding elements in the arrival vector are termed arrival groups. The number of elements within a particular group i is denoted by G_i. The vector consisting of elements G_i is $\mathbf{G}$. The number of buffer groups is given by $|\mathbf{G}|$, where $1 \leq |\mathbf{G}| \leq g$. With arrival vector reduction, instead of generating $\boldsymbol{\mathcal{A}}$ by choosing hs items from $g+1$ addresses with replacement, hs items will be chosen from $|\mathbf{G}|+1$ with replacement.[9] This means that the reduced $\boldsymbol{\mathcal{A}}$ will be a function of the current buffer state $\boldsymbol{\mathcal{B}}_x$. The number chosen from each buffer group is denoted by H_i, where $0 \leq i < |\mathbf{G}|$ and $H_{|\mathbf{G}|}$ is used to denote the number of input channels with no packets (and so not belonging to a buffer group). For each chosen $\mathbf{H}$, $\boldsymbol{\Psi}(H_i, G_i)$ will be generated within each arrival group i, $0 \leq i < |\mathbf{G}|$. Therefore, for each chosen $\mathbf{H}$, $\prod_{i=0}^{|\mathbf{G}|-1} |\boldsymbol{\Psi}(H_i, G_i)|$ arrival vectors will be generated. The set of all these arrival vectors for every possible $\mathbf{H}$ forms the reduced set $\boldsymbol{\mathcal{A}}$. This $\boldsymbol{\mathcal{A}}$ will then be applied to the switch (which is in the given buffer state $\boldsymbol{\mathcal{B}}_x$).

[9] $|\mathbf{G}|$ is typically less than g. For instance, the ground buffer state (empty buffer) has only one buffer group — the group of all the zero elements.

3.3.3.1 Probabilities of Reduced Arrival Vectors

With AVR, both $\mathcal{P}_\alpha$ and $\mathcal{P}_L$ remain unchanged. However, $\mathcal{P}_\delta$ must be revised to account for the fact that a given $\boldsymbol{\mathcal{A}}_y$ no longer contains every possible permutation of addresses within its arrival groups. $\mathcal{P}_\delta$ becomes

$$\mathcal{P}_\delta = \frac{\alpha!}{\mathcal{A}_{y,0}!\mathcal{A}_{y,1}!\cdots\mathcal{A}_{y,g-1}!}\left(\frac{1}{g}\right)^\alpha \varpi. \tag{36}$$

where ϖ accounts for all the possible permutations of the elements within each arrival group. ϖ is given by

$$\varpi = \prod_{i=0}^{|\mathbf{G}|-1} \frac{G_i!}{K_0(i)!K_1(i)!\cdots K_{hs}(i)!}. \tag{37}$$

where $K_j(i)$ is the number of elements with value j in arrival group i. Typically, only a few of the $K_j(i)$ are nonzero. ϖ is the number of unreduced arrival vectors represented by the single, reduced arrival vector $\boldsymbol{\mathcal{A}}_y$.

3.4 Calculation of Switch Performance

Once $\mathbf{P}$ has been constructed, the steady-state probabilities can obtained using Equations 1 and 3. Then, the various performance metrics of the switch, such as the packet loss probability, $\mathcal{P}_{\text{loss}}$, normalized throughput T_h, expected number of packets in the buffer $\mathcal{E}_b$ and the expected packet delay times, can be calculated.

3.4.1 Loss Probability and Throughput

The probability of a packet being lost, $\mathcal{P}_{\text{loss}}$, is defined as

$$\mathcal{P}_{\text{loss}} \equiv \frac{\text{expected number of packets lost per timeslot}}{\text{expected number of input packets per timeslot}}. \tag{38}$$

Because the switch must be in *some* state at the beginning of every timeslot and the states' steady-state probabilities must sum to one, Equation 38 can be rewritten as

$$\mathcal{P}_{\text{loss}} = \frac{\sum_{\boldsymbol{\mathcal{S}}} \pi_x \, \mathcal{E}_{l|x}}{\sum_{\boldsymbol{\mathcal{S}}} \pi_x \, \mathcal{E}_{\alpha|x}}, \tag{39}$$

where $\mathcal{E}_{l|x}$ is the expected number of packets lost per timeslot and $\mathcal{E}_{\alpha|x}$ is the expected number of packets input per timeslot when the switch is in state $\boldsymbol{\mathcal{S}}_x$. $\mathcal{E}_{\alpha|x}$ is a constant with respect to the system states under both RS and ERS traffic. Therefore, $\mathcal{E}_\alpha$, the expected number of arriving packets per timeslot,

$$\mathcal{E}_\alpha = \sum_{\boldsymbol{\mathcal{S}}} \pi_x \, \mathcal{E}_{\alpha|x}, \tag{40}$$

can be found directly for RS,

$$\mathcal{E}_\alpha = hsp, \tag{41}$$

and ERS traffic,

$$\mathcal{E}_\alpha = h \sum_{i=1}^{s} i p_i, \tag{42}$$

where p_i is the probability of an input link having i packets. Therefore, Equation 39 becomes

$$\mathcal{P}_{\text{loss}} = \frac{\sum_{\boldsymbol{\mathcal{S}}} \pi_x \, \mathcal{E}_{l|x}}{\mathcal{E}_\alpha}. \tag{43}$$

For each state $\boldsymbol{\mathcal{S}}_x$, $\mathcal{E}_{l|x}$ is the sum over all possible arrival vectors of $l\mathcal{P}_{\mathcal{A}}$, where l is the number of packets lost from a given arrival vector, and $\mathcal{P}_{\mathcal{A}}$ is the probability of the arrival vector occurring:

$$\mathcal{E}_{l|x} = \sum_{\boldsymbol{\mathcal{A}}} l\mathcal{P}_{\mathcal{A}}. \tag{44}$$

Note that $\mathcal{P}_{\mathcal{A}}$ is a function of the particular arrival vector $\boldsymbol{\mathcal{A}}_y$, and l is dependent on both $\boldsymbol{\mathcal{A}}_y$ and the state $\boldsymbol{\mathcal{S}}_x$. $\mathcal{E}_{l|x}$ can be calculated at the same time as $\mathbf{P}$. The expected loss equivalent of Equation 25 is

$$\mathcal{E}'_{l|x} = \mathcal{E}_{l|x} + l\mathcal{P}_{\mathcal{A}}. \tag{45}$$

Once $\mathcal{P}_{\text{loss}}$ is obtained, the normalized throughput T_h, which is the load the switch presents to its output links, can then be calculated:

$$T_h = \frac{\mathcal{E}_\alpha - \mathcal{E}_\alpha \mathcal{P}_{\text{loss}}}{rg} = \frac{\mathcal{E}_\alpha}{rg}(1 - \mathcal{P}_{\text{loss}}). \tag{46}$$

3.4.2 Buffering and Direct Routing Probabilities

In addition to being dropped, an arriving packet could be sent to the buffer or it could be fortunate enough to make it directly to the switch outputs without passing through the buffer. The probabilities of the latter two cases are denoted by $\mathcal{P}_{\text{buffer}}$ and $\mathcal{P}_{\text{via}}$, respectively. It follows that

$$\mathcal{P}_{\text{loss}} + \mathcal{P}_{\text{buffer}} + \mathcal{P}_{\text{via}} = 1. \tag{47}$$

$\mathcal{P}_{\text{buffer}}$, the probability that an input packet will be sent to the buffer, can be obtained in same manner as $\mathcal{P}_{\text{loss}}$. $\mathcal{P}_{\text{buffer}}$ is defined as

$$\mathcal{P}_{\text{buffer}} \equiv \frac{\text{expected number of input packets buffered per timeslot}}{\text{expected number of input packets per timeslot}}, \tag{48}$$

which can be written as

$$\mathcal{P}_{\text{buffer}} = \frac{\sum_{\boldsymbol{\mathcal{S}}} \pi_x \, \mathcal{E}_{w|x}}{\sum_{\boldsymbol{\mathcal{S}}} \pi_x \, \mathcal{E}_{\alpha|x}}, \tag{49}$$

where $\mathcal{E}_{w|x}$ is the expected number of packets sent to the buffer per timeslot when the switch is in state $\boldsymbol{\mathcal{S}}_x$. Equation 49 can be rewritten using $\mathcal{E}_\alpha$:

$$\mathcal{P}_{\text{buffer}} = \frac{\sum_{\boldsymbol{\mathcal{S}}} \pi_x \, \mathcal{E}_{w|x}}{\mathcal{E}_\alpha}. \tag{50}$$

$\mathcal{E}_{w|x}$ is given by

$$\mathcal{E}_{w|x} = \sum_{\boldsymbol{\mathcal{A}}} w \mathcal{P}_{\mathcal{A}}. \tag{51}$$

w is dependent on both $\boldsymbol{\mathcal{A}}_y$ and the state $\boldsymbol{\mathcal{S}}_x$. Like $\mathcal{E}_{l|x}$, $\mathcal{E}_{w|x}$ can be calculated at the same time as $\mathbf{P}$:

$$\mathcal{E}'_{w|x} = \mathcal{E}_{w|x} + w \mathcal{P}_{\mathcal{A}}. \tag{52}$$

Once $\mathcal{P}_{\text{loss}}$ and $\mathcal{P}_{\text{buffer}}$ are obtained, Equation 47 can be easily solved for $\mathcal{P}_{\text{via}}$.

3.4.3 Expected Number of Packets in the Buffer

The expected number of packets in the buffer, $\mathcal{E}_b$, can be obtained directly from the buffer state names and steady-state probabilities:

$$\mathcal{E}_b = \sum_{\boldsymbol{\mathcal{S}}} \left(\pi_x \sum_{i=0}^{g-1} \mathcal{B}_{x,i} \right). \tag{53}$$

3.4.4 Expected Packet Delay Times

Little's result [67], [68] is an extremely powerful and useful relation. It can be stated as

$$\text{arrival rate} = \frac{\text{expected number of customers in a system}}{\text{expected time spent in the system by each customer}}. \tag{54}$$

Little's result will hold if the number of customers in a system does not grow without bound [49]. Clearly, this is the case for finite-sized OSMA switches.

By having the entire switch serve as the "system," Little's result can be used to obtain the expected "service time," $\mathcal{E}_{t_\mathrm{s}}$, of the incoming packets:

$$\mathcal{E}_{t_\mathrm{s}} = \frac{\mathcal{E}_b}{\mathcal{E}_\alpha}. \tag{55}$$

A packet is considered to be serviced when it is either dropped or routed from the switch.

Typically, one is more interested in $\mathcal{E}_{t_\mathrm{p}}$, the expected time it takes for packets that are not dropped to pass through the switch. In this case, the system arrivals consist only of input packets that have not been dropped. Therefore,

$$\mathcal{E}_{t_\mathrm{p}} = \frac{\mathcal{E}_b}{\mathcal{E}_\alpha(1 - \mathcal{P}_\mathrm{loss})}. \tag{56}$$

Lastly, by defining the system to consist solely of the buffer, the expected amount of time a buffered packet will age in the buffer, $\mathcal{E}_{t_\mathrm{b}}$, can be obtained:

$$\mathcal{E}_{t_\mathrm{b}} = \frac{\mathcal{E}_b}{\mathcal{E}_\alpha \mathcal{P}_\mathrm{buffer}}. \tag{57}$$

$\mathcal{E}_{t_b}$ may be of greater interest with OSMA switches than with ESMP switches because the all-optical buffer of the former can maintain the integrity of its packets only for a limited amount of time. By choosing to route out the oldest buffer packets first, the maximum number of timeslots a packet may remain in the buffer can be limited to m/r (if m is not a multiple of r, round up to the next integer). As mentioned in Section 1.3, operating an OSMA switch in this manner does not otherwise affect switch performance or the model of this work.

3.5 Numerical Results

The full analytic model, as described in the previous and subsequent sections, has been implemented in a computer application, "Shared-optical-Memory Switch Expected Loss calculaToR" (SMELTER). In addition, a discrete event simulator has been developed to provide a means to validate the results from the analytic model. The simulator is known as "Shared-Optical-memory switch Network SIMulator" (SONSIM), and is capable of simulating entire networks of OSMA switches with arbitrary parameters and connection topologies.

The generality of the model provides a large parameter space from which to draw data. As mentioned earlier, this work is primarily concerned with small switches because current all-optical packet switch fabric sizes are severely limited and most existing shared-memory switch models become inaccurate for small switches. Numerical results will now be shown for a chosen subset of parameters to identify the issues and trade-offs associated with evaluating switch performance. Figures 13 and 14 show the packet loss probability and the expected number of packets in the buffer, respectively, for symmetric ($h = g = n = 4,\ s = r = c$) OSMA switches under random traffic ($p = 0.5$), as predicted by both the analytic model and simulation.[10]

[10]Throughout this work, results will be analytic unless stated otherwise. For model validation purposes, simulation results will usually be shown as a nexus of asterisks overlaying the lines of analytic data. The 95% confidence interval of the simulation results is guaranteed to be within the asterisks unless explicit error bars indicate otherwise.

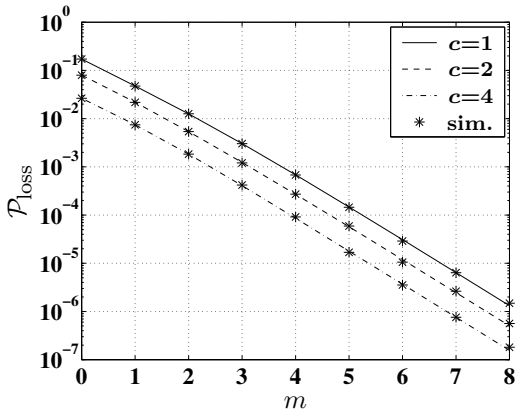


Figure 13: $\mathcal{P}_{\text{loss}}$ for $n = 4$ switches under random traffic with $p = 0.5$.

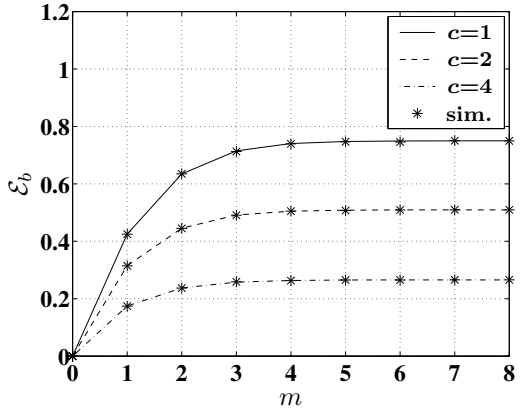


Figure 14: $\mathcal{E}_b$ for $n = 4$ switches under random traffic with $p = 0.5$.

The complete correlation between the analytic and simulation results is unsurprising because of the exact nature of the model.

The first notable observation is the dramatic effect increasing the number of buffer cells has on packet loss. For a switch with the given parameters, each additional buffer cell decreases the probability of a packet being lost by almost an order of magnitude. The expected number of packets in the buffer is perhaps lower than one might expect. These switches operate most of the time with very few, if any, occupied buffer cells. Packet loss can occur only after the buffer saturates — and it is desirable

that this happens quite infrequently. Initially, adding buffer cells increases $\mathcal{E}_b$ because the buffer can hold more packets and the extra cells are used. However, beyond a certain point, the added buffer cells are used so rarely that their presence does not noticeably affect $\mathcal{E}_b$. The result is the characteristic "increasing-then-flat" shape of the curves in Figure 14.

$\mathcal{E}_{t_\mathrm{b}}$, the expected time a buffered packet remains in the buffer, increases with m and decreases with r. For $m \leq r$, $\mathcal{E}_{t_\mathrm{b}} = 1$ (the minimum possible $\mathcal{E}_{t_\mathrm{b}}$) because each buffered packet must remain in the switch for at least one timeslot and all buffered packets are guaranteed to be routed out on the next timeslot if $r \geq m$. For the switches in Figures 13 and 14, the largest $\mathcal{E}_{t_\mathrm{b}}$ of ≈ 1.28 occurs for the $c = 1, m = 8$ switch. OSMA switches are very efficient at clearing their buffers and achieve theoretically minimum $\mathcal{E}_{t_\mathrm{b}}$ values.

The effect of channel grouping is somewhat more complex. Clearly, increasing r decreases $\mathcal{P}_{\mathrm{loss}}$ because of the reduction in output port contention. However, because the input link load remains constant at 50%, the $c = 4$ switch is carrying four times the bandwidth of the $c = 1$ switch! This raises the issue of whether Figure 13 is a fair comparison. The answer depends on what one hopes to achieve by utilizing channel grouping. If it is viewed as a means to increase switch capacity, then Figure 13 is quite appropriate. On the other hand, if channel grouping is seen primarily as a way to reduce packet loss while maintaining the existing switched bandwidth then the comparison will need modification. The rational for this latter position is that all-optical networks may have some "bandwidth to burn" in exchange for reduced optical buffer size requirements (i.e., switch/buffer cost may be traded for link cost). The model of this work can be used to explore either approach. Figure 15 shows $\mathcal{P}_{\mathrm{loss}}$ for the same switch as in Figure 13 except that the total switched bandwidth is conserved (input link load $= \frac{0.5}{c}$). Note the dramatic decrease in $\mathcal{P}_{\mathrm{loss}}$ obtained via channel grouping as compared to that of Figure 13.

$\mathcal{P}_{\mathrm{loss}}$ and capacity benefits aside, channel grouping is undesirable because it man-

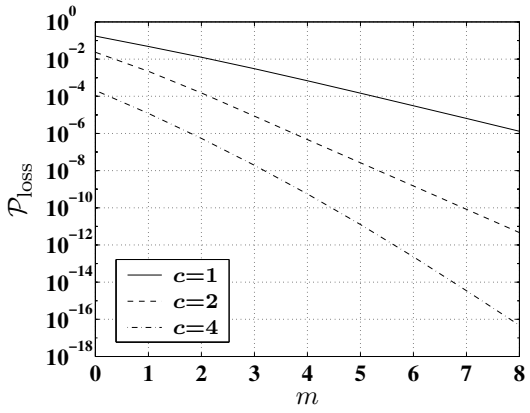


Figure 15: $\mathcal{P}_{\text{loss}}$ for $n = 4$ switches under random traffic with $p = \frac{0.5}{c}$.

dates an increase in the complexity of the switch fabric and the network interconnects. It is particularly unwanted at the switch inputs because it increases the maximum number of packets that can arrive at any given timeslot. In this way, input channel grouping serves to undermine some of the $\mathcal{P}_{\text{loss}}$ benefit of channel grouping at the outputs. This considered, when $\mathcal{P}_{\text{loss}}$ is of primary concern it is desirable to provide channel grouping only at the outputs. However, these channel grouped outputs will have to be connected to *something* — be it an end user or the input to the next stage of a network. Symmetric switches allow for easy, direct, interconnections between stages without the unrestrained growth of channel grouping factors as one progresses through the network. Therefore, the two classes of channel-grouped switches of greatest interest are (1) those using channel grouping only at the outputs — such switches may be used in the first stage of a network; and (2) those using symmetric channel grouping — these can be employed in any stage(s) of a network. If channel grouping is confined to the outputs, the result is shown in Figure 16. (Not shown is the $r = 4$ case in which output port contention is impossible and even bufferless switches are *lossless* under all loads.) Without the negative effect of input channel grouping, and with all of the benefits of output channel grouping, this configuration has the lowest $\mathcal{P}_{\text{loss}}$ of all the given examples.

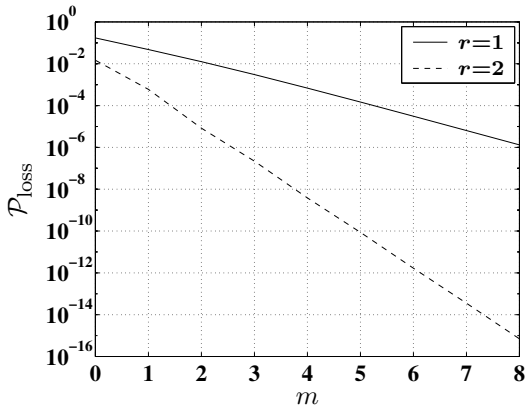


Figure 16: $\mathcal{P}_{\text{loss}}$ for $n = 4$ switches under random traffic with $p = 0.5$, $s = 1$, $r = 1, 2$.

Channel-grouped switches should be evaluated against alternatives such as increasing the channel capacity (speed-up the data rate so that the input load decreases) or adding buffer cells. The cost of implementing each alternative plays a big role in determining which is best. For the example switch, if channel grouping is not used, but instead the link (channel) capacity is increased so that the input link load decreases ($p = 0.5, \frac{0.5}{2}, \frac{0.5}{4}$), $\mathcal{P}_{\text{loss}}$ becomes as shown in Figure 17. The result is somewhat similar to that achieved by using channel grouping with constant total switched bandwidth in Figure 15. However, the cost to speed up a single link by c may be very different

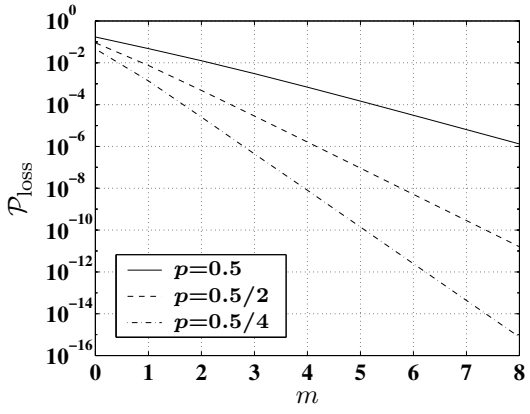


Figure 17: $\mathcal{P}_{\text{loss}}$ for $n = 4$, $c = 1$ switches under random traffic with $p = 0.5, \frac{0.5}{2}, \frac{0.5}{4}$.

from the cost to provide c parallel channels within a link. Thus, the model can be used to explore design options.

Figure 17 shows another important aspect of OSMA switches — the lighter the load, the greater the decrease in $\mathcal{P}_{\text{loss}}$ each additional buffer cell provides. This is because lighter loads produce less correlation between the buffer cells as well as lower $\mathcal{E}_b$ values (because the buffer has a better chance to clear itself at each timeslot). Thus, for lighter loads, each additional buffer cell produces a greater benefit than it would under higher loads. So, if low $\mathcal{P}_{\text{loss}}$ is important, switches with limited buffers are likely to be constrained to operation under light loads.

Finally, Figure 18 shows that $\mathcal{P}_{\text{loss}}$ is very sensitive to changes in the input load.[11]

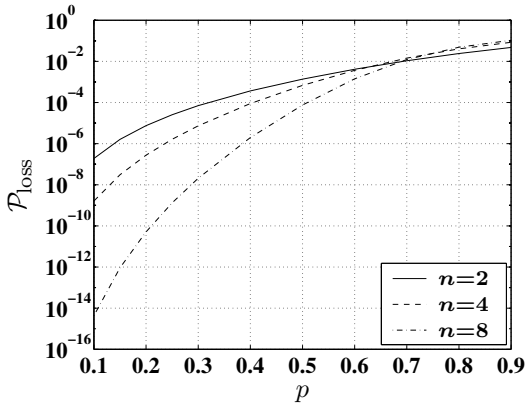


Figure 18: $\mathcal{P}_{\text{loss}}$ for $m = n$, $c = 1$ switches under random traffic of various loads.

For a given number of buffer cells per input/output line (m/n) under reasonable loads, larger switches have lower loss rates because of lower correlation between the addresses of the packets in the buffer cells.[12] However, as the load becomes very heavy, the buffer

[11] For plots with a continuous x-axis variable (e.g., p, σ), data points are guaranteed to be plotted for each x-axis tick mark value with additional data points as needed to prevent excessive interpolation between data points (e.g., $p = 0.15, 0.25$ in this particular figure). Obviously, this interpolation of the graphing software is not a concern with discrete x-axis variables (e.g., m).

[12] This independence of the addresses of the buffered packets is a key assumption of Karol's approximate model which works well for large switches (Section 2.1).

saturates to a degree that it becomes virtually useless, and $\mathcal{P}_{\text{loss}}$ is not much better than that of a bufferless switch. Under such extreme conditions, the larger switches exhibit a slightly higher loss rate (this is also true for bufferless switches under the same conditions).

Chapter 4

OSMA Switches Under Bursty Traffic

End users tend to send their data as quickly as possible. When contention occurs, switch buffers allow the build-up of groups of packets destined for the same output link. For these reasons, real-world network traffic often has correlations between the arrivals of packets in a link. So, packets tend to arrive in "clumps" or "bursts." For instance, if there is an arriving packet in a link in a given timeslot, then there often is a greater than average probability that a packet will arrive in the next timeslot(s). Such temporally bursty behavior implies that the data source has memory and is not accurately represented by the RS and ERS models of Chapter 3; a more complex traffic source model is required.

4.1 The Two-State Source (TS)

The Markov model for a two-state bursty source is shown in Figure 19. This source's

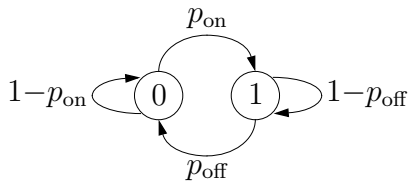


Figure 19: The two-state source.

properties are determined by the parameters p_{on}, p_{off} and p. When in the off state (0), no packets will be emitted. When in the on state (1), a packet will be emitted with probability p. The transition probabilities between the two states are given by p_{on} and p_{off}. When $p = 1$, the source will always emit packets when in the on state. Any bursty source that uses all of the available bandwidth when on will be referred to

as a "full-on-burst" (FOB) source. The steady-state probability of finding the source in the on state, π_{on}, is given by

$$\pi_{\text{on}} = \frac{p_{\text{on}}}{p_{\text{on}} + p_{\text{off}}}. \tag{58}$$

The steady-state probability for the off state is given by

$$\pi_{\text{off}} = \frac{p_{\text{off}}}{p_{\text{on}} + p_{\text{off}}}. \tag{59}$$

The average load presented by the source, σ, is given by

$$\sigma = \pi_{\text{on}} p. \tag{60}$$

The average burst length[1], λ, can be obtained from

$$\lambda = \frac{1}{p_{\text{off}}}. \tag{61}$$

λ is equal to the expected number of consecutive packets emitted only for a FOB source ($p = 1$).

Note that, if p_{off} and p_{on} are taken to the extremes of zero and one, the two-state source may become periodic or deterministic. If so, it will be incapable of "forgetting" its initial starting state.[2] Care must be taken when at the boundaries of the parameter space to ensure that the choice of parameters does not violate the requirements of a valid Markov chain.

For the purposes of this work, the address of each packet in a burst is uniformly and randomly chosen from the possible destination addresses. Hence, the addresses of the packets within a burst or between bursts are uncorrelated. Although this may not be a completely realistic model of end-user bursts, it may accurately reflect the situation where burstiness is the result of switch buffering or where network traffic is the result of the superposition of packets from a large number of end users [59]. The

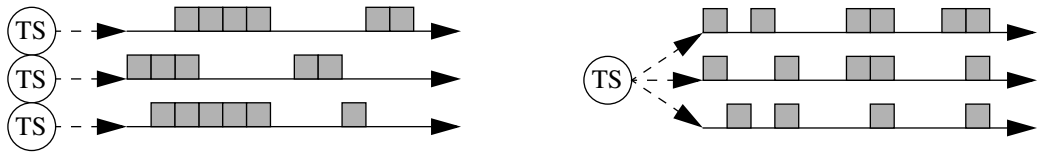
[1] A source is said to "burst" when it is in the on state for one or more consecutive timeslots.

[2] Consider $p = p_{\text{on}} = p_{\text{off}} = 1$. This source will then emit a packet *every* other timeslot like a "square wave" generator.

high capacity of the OSMA switch is likely to be used at the core of a high bandwidth all-optical network where the traffic is the composite of many, perhaps thousands or even millions, of simultaneous lower bandwidth end users. In such a situation, the traffic presented to a switch can be less bursty than that from the individual end users [69], [70].

4.1.1 The Extended Two-State Source (ETS)

As discussed in Section 3.1.1, a source can be interfaced to a multichannel link by driving each channel independently or by speeding up a single source to drive the entire link (Figure 20). Because the two-state source has memory, the two approaches



(a) Drive independently. (b) Speed-up a single source.

Figure 20: Interfacing a two-state source to a multichannel link.

do *not* yield identical traffic patterns. There is a problem with analyzing switches driven by "sped-up" multistate sources — the source can change state in the middle of "filling" a link (Figure 20(b)). For this reason, the sped-up source will be allowed to change states at "link" timeslots only. This restriction will ensure that the state transitions of the switch and sources will always occur at the same time.

The occupancy probabilities for the independently driven link are given by

$$q_i = \binom{c}{i} (\pi_{\text{on}} p)^i (1 - \pi_{\text{on}} p)^{c-i}. \tag{62}$$

For the sped-up source, q_i is given by

$$q_i = \begin{cases} \pi_{\text{off}} + \pi_{\text{on}}(1-p)^c & : \quad i = 0 \\ \pi_{\text{on}} \binom{c}{i} p^i (1-p)^{c-i} & : \quad i \neq 0 \end{cases}. \tag{63}$$

The average load presented to the link, for both the independently driven and sped-up cases, is given by

$$\sigma = \pi_{\text{on}} p. \tag{64}$$

A more general form of the sped-up two-state source, the extended two-state (ETS) source, can be had by allowing an arbitrary probability distribution for the number of packets emitted per timeslot when the source is on. Hence, when on, the ETS source will emit $0, 1, \ldots, c$ packets into the link with probabilities $p_0, p_1, \ldots, p_c$, respectively, where $\sum_{i=0}^{c} p_i = 1$. If $p_c = 1$, the ETS source is classified as a FOB source. The link occupancy probabilities for the ETS source are given by

$$q_i = \begin{cases} \pi_{\text{off}} + \pi_{\text{on}} p_0 & : \quad i = 0 \\ \pi_{\text{on}} p_i & : \quad i \neq 0 \end{cases}. \tag{65}$$

The load presented to the link by the ETS source is

$$\sigma = \frac{\sum_{i=1}^{c} i q_i}{c} = \frac{\pi_{\text{on}} \sum_{i=1}^{c} i p_i}{c}. \tag{66}$$

Another property of interest for bursty sources is the average load presented when on, σ_{on}. For the independently driven and sped-up cases, σ_{on} is given by

$$\sigma_{\text{on}} = p. \tag{67}$$

Note that, for the independently driven link, σ_{on} refers to the *channel* load presented when the source is on (there is one source per channel). The entire link will have this load only when *all* of the link's sources are on. For the ETS source, σ_{on} is given by

$$\sigma_{\text{on}} = \frac{\sum_{i=1}^{c} i p_i}{c}. \tag{68}$$

In any event, because the bursty sources are not allowed to emit packets when off, it is always true that $\sigma \leq \sigma_{\text{on}}$.

4.1.2 The Temporal-Burstiness Factor

It is helpful to have a quantitative measure of the temporal burstiness of traffic. The temporal-burstiness factor, β, is defined in this work as

$$\beta \equiv \frac{\begin{array}{c}\text{the probability that there will be one or more packets in the link in the next timeslot,} \\ \text{given that one or more packets are in the link in this timeslot}\end{array}}{\text{the probability that there will be one or more packets in the link}}. \quad (69)$$

$\beta = 1$ indicates a source that is not temporally bursty, such as a random source. $\beta < 1$ indicates a "relaxation" or antibursty source, which has a tendency not to emit back-to-back packets. $\beta > 1$ indicates bursty traffic.

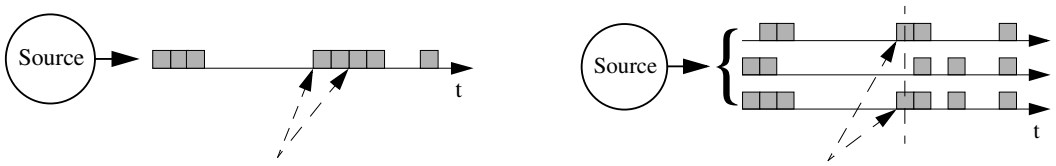
β may be determined directly from empirical traffic data using Equation 69. For a two-state source (including the multichannel sped-up and ETS forms), β is given by

$$\beta = \frac{1 - p_{\text{off}}}{\pi_{\text{on}}}. \quad (70)$$

It should be pointed out that as the average load of the bursty source, σ, increases, the upper bound on β decreases via constraints on π_{on} and p. Therefore, sources of higher loads cannot be made as bursty as lighter load sources — a source that is putting out packets almost all the time can hardly burst to an even higher emission level. It may also be surprising to note that, for a large enough p_{off} (or π_{on}), β can fall below unity. Hence, the two-state source model is also capable of generating antibursty traffic.

4.2 Spatial Burstiness

In addition to temporal burstiness, channel-grouped systems can be subject to another type of burstiness which is called "spatial burstiness" in this work. Temporal burstiness is the tendency of packets to arrive in successive timeslots, but spatial burstiness is the tendency of packets to arrive together in the *same* timeslot (Figure 21). Although different, these two types of burstiness are related. For instance, it is easy to see that if a temporally bursty source is sped-up to drive a multichannel link, the resulting traffic will be spatially bursty (Figure 20(b)). However, unlike



(a) Temporal burstiness. (b) Spatial burstiness.

Figure 21: Temporal versus spatial burstiness.

temporal burstiness, spatial burstiness can occur even in memoryless systems. For example, a bufferless switch under random input traffic can produce spatially bursty output traffic. This is the direct result of the multiple output channels being used to resolve what would otherwise be output link contention. Both the ERS and ETS sources can produce spatially bursty traffic through a suitable choice of p_i.

The reader should note that most publications in the area of switching theory use the term "bursty" to refer exclusively to temporally bursty traffic. In this work, the term "bursty" will be qualified when the distinction is important.

4.2.1 The Spatial-Burstiness Factor

As with temporal burstiness, it is helpful to have a quantitative measure of the spatial burstiness of traffic. The spatial-burstiness factor, β_S, is defined in this work as

$$\beta_S \equiv \frac{\text{the expected number of packets in the link, given that one or more packets are in the link}}{\text{the expected number of packets in the link, given that one or more packets are in the link, for traffic with a binomial occupancy distribution } (q_i) \text{ of the same average load}}. \tag{71}$$

Thus, if the channels of a link behave independently of each other (binomial distribution of the same average load), even if the traffic in the link is temporally bursty, then $\beta_S = 1$ which indicates non-spatially bursty traffic. Spatial burstiness occurs when there is a positive correlation in the presence of packets in each channel of a link ($\beta_S > 1$).

β_S may be determined directly from q_i obtained from empirical traffic data or

from the parameters of a traffic model using Equation 71 written in the form

$$\beta_S = \frac{\frac{\sum_{i=1}^{c} iq_i}{\sum_{i=1}^{c} q_i}}{\frac{c\sigma}{1-(1-\sigma)^c}}. \tag{72}$$

For the sped-up TS source, Equation 72 can be written as

$$\beta_S = \frac{\frac{cp}{1-(1-p)^c}}{\frac{c\sigma}{1-(1-\sigma)^c}} = \frac{p(1-(1-\sigma)^c)}{\sigma(1-(1-p)^c)}. \tag{73}$$

For the ETS sources, Equation 72 simplifies to

$$\beta_S = \frac{1-(1-\sigma)^c}{\pi_{\text{on}}(1-p_0)}. \tag{74}$$

Finally, β_S for ERS sources can be obtained using

$$\beta_S = \frac{1-(1-\sigma)^c}{1-p_0}, \tag{75}$$

which is just Equation 74 with $\pi_{\text{on}} = 1$.

4.3 System States Under Bursty Traffic

The buffer state of a switch under temporally bursty traffic does not contain all of the historical dependencies of the switch/source system because one also needs to know the states of the input sources to predict the probability of an arrival vector and, consequently, the probabilities of the buffer state transitions the given arrival vector may cause. Therefore, to be a valid Markov model, the system state space, $\boldsymbol{\mathcal{S}}$, must be extended to include the buffer states, $\boldsymbol{\mathcal{B}}$, as well as the state space of the input sources, $\boldsymbol{\mathcal{I}}$. Hence, $\boldsymbol{\mathcal{S}}$ becomes

$$\boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{B}} \otimes \boldsymbol{\mathcal{I}}. \tag{76}$$

The $\otimes$ operator forms a state set by prepending each element in the set on the left hand side to every element in the set on the right hand side. Therefore, $|\boldsymbol{\mathcal{S}}| = |\boldsymbol{\mathcal{B}}||\boldsymbol{\mathcal{I}}|$. For example, if $\boldsymbol{\mathcal{B}} = \{[0,0,0,0],\ [1,0,0,0]\}$ and $\boldsymbol{\mathcal{I}} = \{0,1\}$, then

$$\boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{B}} \otimes \boldsymbol{\mathcal{I}} = \{[0,0,0,0].0,\ [0,0,0,0].1,\ [1,0,0,0].0,\ [1,0,0,0].1\},$$

where "." is used to separate the constituent parts of the composite states. $\boldsymbol{\mathcal{S}}_x^{\mathcal{B}}$ and $\boldsymbol{\mathcal{S}}_x^{\mathcal{I}}$ will refer to the buffer-state part and the input-state part of system state $\boldsymbol{\mathcal{S}}_x$, respectively ($\boldsymbol{\mathcal{S}}_x = \boldsymbol{\mathcal{S}}_x^{\mathcal{B}}.\boldsymbol{\mathcal{S}}_x^{\mathcal{I}}$). As before, x will be used to index system states. Additionally, $x_{\mathcal{B}}$ and $x_{\mathcal{I}}$ will be used as needed to index buffer ($\boldsymbol{\mathcal{B}}_{x_{\mathcal{B}}}$) and input ($\boldsymbol{\mathcal{I}}_{x_{\mathcal{I}}}$) states, respectively.

4.3.1 General Input Source State Description

One way to represent the states of the input sources is by using a binary vector whose elements indicate the status of each input source. If I denotes the number of input sources, then $|\boldsymbol{\mathcal{I}}|$ is given by

$$|\boldsymbol{\mathcal{I}}| = 2^I. \tag{77}$$

Unfortunately, $|\boldsymbol{\mathcal{I}}|$ grows much too rapidly for OSMA switch sizes of interest. However, for completeness, the probability of transitioning between two input states ($\mathcal{P}_{\mathcal{T}_{\mathcal{I}}}$) in this $\boldsymbol{\mathcal{I}}$ will now be developed.

4.3.1.1 General Input Source State Transition Probabilities

The probability of transitioning from input state $\boldsymbol{\mathcal{I}}_{x_{\mathcal{I}}}$ to $\boldsymbol{\mathcal{I}}_{f_{\mathcal{I}}}$ is given by

$$\mathcal{P}_{\mathcal{T}_{\mathcal{I}}} = \prod_{i=0}^{I-1} \varrho(i), \tag{78}$$

where

$$\varrho(i) = \begin{cases} 1 - p_{\text{on}} & : \quad \mathcal{I}_{x_{\mathcal{I}},i} = 0,\ \mathcal{I}_{f_{\mathcal{I}},i} = 0 \\ p_{\text{on}} & : \quad \mathcal{I}_{x_{\mathcal{I}},i} = 0,\ \mathcal{I}_{f_{\mathcal{I}},i} = 1 \\ p_{\text{off}} & : \quad \mathcal{I}_{x_{\mathcal{I}},i} = 1,\ \mathcal{I}_{f_{\mathcal{I}},i} = 0 \\ 1 - p_{\text{off}} & : \quad \mathcal{I}_{x_{\mathcal{I}},i} = 1,\ \mathcal{I}_{f_{\mathcal{I}},i} = 1 \end{cases}, \tag{79}$$

which can be seen directly from Figure 19.

4.3.2 Reduced Input Source State Description

For the purposes of calculating the probability of a given arrival vector $\boldsymbol{\mathcal{A}}_y$, it does not matter which particular input source is in which particular state. It is the total number of on-sources which affects $\mathcal{P}_{\mathcal{A}}$. This is a direct result of the switch model being indifferent to which inputs supply which packets.

Therefore, $\boldsymbol{\mathcal{I}}$ can be reduced to a set of scalar values that indicate how many sources are on:

$$\boldsymbol{\mathcal{I}} = \{0, 1, 2, \ldots, I\} \tag{80}$$

and

$$|\boldsymbol{\mathcal{I}}| = I + 1. \tag{81}$$

In this way, $|\boldsymbol{\mathcal{S}}| = |\boldsymbol{\mathcal{B}}||\boldsymbol{\mathcal{I}}|$ grows linearly with an increasing number of bursty input sources and is only a factor of $I+1$ larger than that of the same switch under random traffic.

For the reduced input source state description, $\boldsymbol{\mathcal{S}}_x^{\mathcal{I}}$, $\boldsymbol{\mathcal{I}}_{x_{\mathcal{I}}}$ and $\boldsymbol{\mathcal{I}}_{f_{\mathcal{I}}}$ are scalars and can be written as $\mathcal{S}_x^{\mathcal{I}}$, $\mathcal{I}_{x_{\mathcal{I}}}$ and $\mathcal{I}_{f_{\mathcal{I}}}$, respectively. Unless otherwise noted, the reduced input source state description will be used exclusively in the remainder of this work. Also, note that the $\mathcal{S}_x^{\mathcal{I}}$ representation differs from that of $\mathcal{I}_{x_{\mathcal{I}}}$ only in the way the states are numbered (indexed). So, either can be used depending upon whether the system (x, f) or just the input $(x_{\mathcal{I}}, f_{\mathcal{I}})$ states need to be counted.

4.3.2.1 Reduced Input Source State Transition Probabilities

Calculating $\mathcal{P}_{\mathcal{I}_{\mathcal{I}}}$ with the reduced $\boldsymbol{\mathcal{I}}$ is not as easy as that of the nonreduced case in Section 4.3.1.1. The probability of transitioning from input state $\mathcal{I}_{x_{\mathcal{I}}}$ to $\mathcal{I}_{f_{\mathcal{I}}}$ is given by

$$\mathcal{P}_{\mathcal{I}_{\mathcal{I}}} = \sum_{a=\text{lower}(a)}^{\text{upper}(a)} \Bigg(\Big((1-p_{\text{off}})^a (p_{\text{off}})^{\mathcal{I}_{x_{\mathcal{I}}}-a} (p_{\text{on}})^{\mathcal{I}_{f_{\mathcal{I}}}-a} (1-p_{\text{on}})^{I-\mathcal{I}_{x_{\mathcal{I}}}-\mathcal{I}_{f_{\mathcal{I}}}+a} \Big) \cdot \frac{\mathcal{I}_{x_{\mathcal{I}}}!}{a!(\mathcal{I}_{x_{\mathcal{I}}}-a)!} \frac{(I-\mathcal{I}_{x_{\mathcal{I}}})!}{(\mathcal{I}_{f_{\mathcal{I}}}-a)!(I-\mathcal{I}_{x_{\mathcal{I}}}-\mathcal{I}_{f_{\mathcal{I}}}+a)!} \Bigg), \tag{82}$$

where a is the number of on-sources that remain on between the two states and

$$\text{lower}(a) = \max(\mathcal{I}_{x_\mathcal{I}} + \mathcal{I}_{f_\mathcal{I}} - I,\ 0) \tag{83}$$

$$\text{upper}(a) = \min(\mathcal{I}_{x_\mathcal{I}},\ \mathcal{I}_{f_\mathcal{I}}). \tag{84}$$

Equations 82, 83 and 84 merit further discussion. There are four categories of things that can happen to the sources when they change states. Knowing the number of on-sources in the initial state, $\mathcal{I}_{x_\mathcal{I}}$, the number of sources that are on after the transition, $\mathcal{I}_{f_\mathcal{I}}$, and the number of sources that remained on through the transition, a, provides one with enough information to determine how many sources belong to each of the four categories:

$$\begin{aligned} a &: \text{ sources that remained on} \\ \mathcal{I}_{x_\mathcal{I}} - a &: \text{ sources that were on that went off} \\ \mathcal{I}_{f_\mathcal{I}} - a &: \text{ sources that were off that went on} \\ I - \mathcal{I}_{x_\mathcal{I}} - \mathcal{I}_{f_\mathcal{I}} + a &: \text{ sources that were off that stayed off} \end{aligned}$$

The four exponentials in Equation 82 calculate the probability that the sources in each category would transition in the way that they did. However, there is usually more than a single way in which the sources can transition that will result in the same number of sources in each category. The first fraction enumerates the number of ways to choose a sources from the $\mathcal{I}_{x_\mathcal{I}}$ that are on to remain so. The second fraction enumerates the number of ways to choose the $\mathcal{I}_{f_\mathcal{I}} - a$ sources that went from off to on from those that were initially off. There is no need to enumerate the on–off and off–off cases as they have been already counted — choosing sources to remain on implies that the unchosen go off.

Finally, the probability of transitioning for the given a must be summed over all possible values of a. The valid range of a is determined as follows: Obviously, a cannot be larger than either $\mathcal{I}_{x_\mathcal{I}}$ or $\mathcal{I}_{f_\mathcal{I}}$. The minimum value of a can be zero, unless the number of on-sources in the next state, $\mathcal{I}_{f_\mathcal{I}}$, exceeds the number of off states in the initial state, $(\mathcal{I}_{f_\mathcal{I}} > I - \mathcal{I}_{x_\mathcal{I}})$. This situation means that the number of on-sources in the next state of this transition is so large that, even if *every* off-source in the

initial state turned on, there would not be enough to account for $\mathcal{I}_{f_\mathcal{I}}$ — there has to be at least $\mathcal{I}_{f_\mathcal{I}} + \mathcal{I}_{x_\mathcal{I}} - I$ sources that remain on (a has a floor). Therefore, the valid range of a is given by Equations 83 and 84.

4.4 System State Transition Probabilities

Under bursty traffic, the probability of a particular transition between system states ($\mathcal{P}_\mathcal{T}$) occurring, in the manner determined by the arrival and loss vectors, is given by

$$\mathcal{P}_\mathcal{T} = \mathcal{P}_{\mathcal{T}_\mathcal{I}} \mathcal{P}_{\mathcal{T}_\mathcal{B}} = \mathcal{P}_{\mathcal{T}_\mathcal{I}} \mathcal{P}_\alpha \mathcal{P}_\delta \mathcal{P}_L, \tag{85}$$

which is Equation 24 after having been extended to encompass the transitions of the input sources. $\mathcal{P}_\delta$ and $\mathcal{P}_L$ remain the same as under random traffic because the input states do not affect the addresses of the packets or the internal operation of the switch. However, $\mathcal{P}_\alpha$ must be modified to account for the fact that only sources that are on (active) can emit packets. Therefore, $\mathcal{P}_\alpha$ is a function of the input state. $\mathcal{P}_\alpha$ will be derived in the next sections.

4.4.1 $\mathcal{P}_\alpha$ With Each Channel Driven Independently

If each input channel is driven by a separate two-state bursty source, Equation 30 can be modified to give

$$\mathcal{P}_\alpha = \begin{cases} \frac{\mathcal{S}_x^\mathcal{I}!}{\alpha!(\mathcal{S}_x^\mathcal{I}-\alpha)!} p^\alpha (1-p)^{\mathcal{S}_x^\mathcal{I}-\alpha} & : \quad 0 \le \alpha \le \mathcal{S}_x^\mathcal{I} \\ 0 & : \quad \text{otherwise} \end{cases}, \tag{86}$$

as only the number of active input sources, $\mathcal{S}_x^\mathcal{I}$, can contribute arriving packets.

4.4.2 $\mathcal{P}_\alpha$ For Sped-Up TS Source Driven Links

If the input links are driven by sped-up two-state sources, $\mathcal{P}_\alpha$ is given by

$$\mathcal{P}_\alpha = \begin{cases} \frac{(\mathcal{S}_x^\mathcal{I} s)!}{\alpha!(\mathcal{S}_x^\mathcal{I} s-\alpha)!} p^\alpha (1-p)^{\mathcal{S}_x^\mathcal{I} s-\alpha} & : \quad 0 \le \alpha \le \mathcal{S}_x^\mathcal{I} s \\ 0 & : \quad \text{otherwise} \end{cases}, \tag{87}$$

as only the number of active input sources, $\mathcal{S}_x^\mathcal{I}$, can contribute arriving packets.

4.4.3 $\mathcal{P}_\alpha$ Under ETS Traffic

If the input links are driven by ETS sources, Equation 32 can be modified for bursty traffic to give

$$\mathcal{P}_\alpha = \begin{cases} \sum\limits_{\mathbf{\Psi}(\alpha,\mathcal{S}_x^{\mathcal{I}},s)} \left(\frac{\mathcal{S}_x^{\mathcal{I}}!}{K_0!K_1!\cdots K_s!} p_0^{K_0} p_1^{K_1} \cdots p_s^{K_s} \right) & : \quad 0 \leq \alpha \leq \mathcal{S}_x^{\mathcal{I}} s \\ 0 & : \quad \text{otherwise} \end{cases}, \tag{88}$$

as only the number of active input sources, $\mathcal{S}_x^{\mathcal{I}}$, can contribute arriving packets.

4.5 Calculation of Switch Performance

The few remaining changes needed to extend the model to handle bursty traffic will now be reviewed.

Because there are only $I + 1$ elements in $\boldsymbol{\mathcal{I}}$, $\mathcal{P}_{\mathcal{I}_\mathcal{I}}$ can easily be precomputed and stored in a relatively small lookup matrix with $(I + 1)^2$ elements. Therefore, in practice, $\mathcal{P}_{\mathcal{I}_\mathcal{I}}$ can usually be obtained repeatedly with less effort than $\mathcal{P}_{\mathcal{I}_\mathcal{B}}$. For this reason, when calculating $\mathcal{P}_\mathcal{I}$, it is usually more efficient to calculate $\mathcal{P}_{\mathcal{I}_\mathcal{B}}$ for each buffer-state/arrival-vector and then to evaluate $\mathcal{P}_{\mathcal{I}_\mathcal{I}}$ for each possible input-state transition rather than the other way around. Equations 25, 45 and 52 can then be used to obtain $\mathbf{P}$, $\mathcal{E}_{l|x}$ and $\mathcal{E}_{w|x}$, respectively.

Under bursty traffic, $\mathcal{E}_{\alpha|x}$ is not a constant with respect to the system states because it is a function of the input state. If each input channel is driven independently, $\mathcal{E}_{\alpha|x}$ is given by

$$\mathcal{E}_{\alpha|x} = \mathcal{S}_x^{\mathcal{I}} p. \tag{89}$$

For ETS inputs, it is given by

$$\mathcal{E}_{\alpha|x} = \mathcal{S}_x^{\mathcal{I}} \sum_{i=1}^{s} i p_i, \tag{90}$$

which can also be used with the sped-up source by using p_i that are binomial coefficients.

The expected number of packets in the buffer, $\mathcal{E}_b$, can be obtained like in Equation 53, but noting that the buffer state under each system state is now given by $\boldsymbol{\mathcal{S}}_x^{\mathcal{B}}$ rather than by $\boldsymbol{\mathcal{B}}_x$ as with random traffic:

$$\mathcal{E}_b = \sum_{\boldsymbol{\mathcal{S}}} \left(\pi_x \sum_{i=0}^{g-1} \mathcal{S}_{x,i}^{\mathcal{B}} \right). \tag{91}$$

4.6 Model Algorithm Overview

It may be helpful to summarize the steps required by the algorithm of the model of this work as developed in the previous sections. There are three major steps:

1. Calculate the transition probability matrix ($\mathbf{P}$), the expected number of packets lost in each timeslot for each state ($\mathcal{E}_{l|x}$) and the expected number of packets sent to the buffer in each timeslot for each state ($\mathcal{E}_{w|x}$) using a series of nested loops that go through each possible buffer state, arrival vector, way of dropping packets (if needed), input state and input state transition. The probability of each transition is calculated and then added into $\mathbf{P}$:

```
For each buffer state
  For each arrival vector
    For each way of dropping packets
      For each possible input state[3]
        Build E_{l|x} and E_{w|x} using Equations 45 and 52.
        Build E_{links,i|x} using Equation 97.[4]
        For each possible input state transition
          Calculate the transition probability to the next system state
          (P_T = P_A P_L P_{T_I}) and enter it into P.
```

[3] Because calculating arrival vectors is relatively computationally intensive, it is more efficient to put calculations that depend on the input state at the inner part of the loop. There, the set of possible input states is restricted by the properties of the current arrival vector (because some input states cannot produce some arrival vectors). This is one of several acceleration techniques employed by SMELTER.

[4] $\mathcal{E}_{\text{links},i|x}$ will be discussed in Section 5.3.

2. Solve for the steady-state probabilities, $\boldsymbol{\pi}$, via the iteration of Equation 1.

3. Finally, using $\boldsymbol{\pi}$, $\mathcal{E}_{l|x}$ and $\mathcal{E}_{w|x}$, calculate the performance results (overall loss rate, average number of packets in the switch, average delay, etc.) using the equations of Sections 3.4 and 4.5.

4.7 Scalability Issues

There are two main issues that determine if the model is computationally tractable for a given switch. The first, and usually the most important, is whether the transition matrix $\mathbf{P}$ will fit into available computer memory. $|\mathbf{P}|$ scales as the number of system states squared ($|\boldsymbol{\mathcal{S}}|^2$) with about 10 bytes needed per element for floating point representation.

The second issue is the number of calculations that are required to generate $\mathbf{P}$. This is at least $|\boldsymbol{\mathcal{S}}||\boldsymbol{\mathcal{A}}|$.[5] The required number of computations is normally not of as much importance as the memory requirement. Usually, the only time the number of computations becomes a limiting factor is for large switches with small buffers.[6]

There is also the number of computations required to solve for the steady-state probabilities. This is approximately $2|\boldsymbol{\mathcal{S}}|^2$ multiplied by the number of required iterations (typically around 50–400). If $\mathbf{P}$ will fit into available computer memory, solving for $\boldsymbol{\pi}$ is normally not too much of an issue as most current computers can perform several hundred operations on each of their memory cells within a reasonable amount of time (i.e., the amount of memory is reasonably matched to the processing speed).

It is interesting to compare the state space size requirements of the model of this work (called here the "partition" model) with that of the closest ESMP model — Pattavina's 3-D approximate model [24]. For the 3-D model, $|\boldsymbol{\mathcal{S}}|$ is given by

[5] In practice, the number of computations can be more than an order of magnitude greater than $|\boldsymbol{\mathcal{S}}||\boldsymbol{\mathcal{A}}|$ because the number of ways a switch can drop packets for a given $\boldsymbol{\mathcal{S}}_x$, $\boldsymbol{\mathcal{A}}_y$ (Section 4.6) is not considered. Also, there is programming "overhead" associated with generating each $\boldsymbol{\mathcal{A}}_y$ and $\mathbf{L}$.

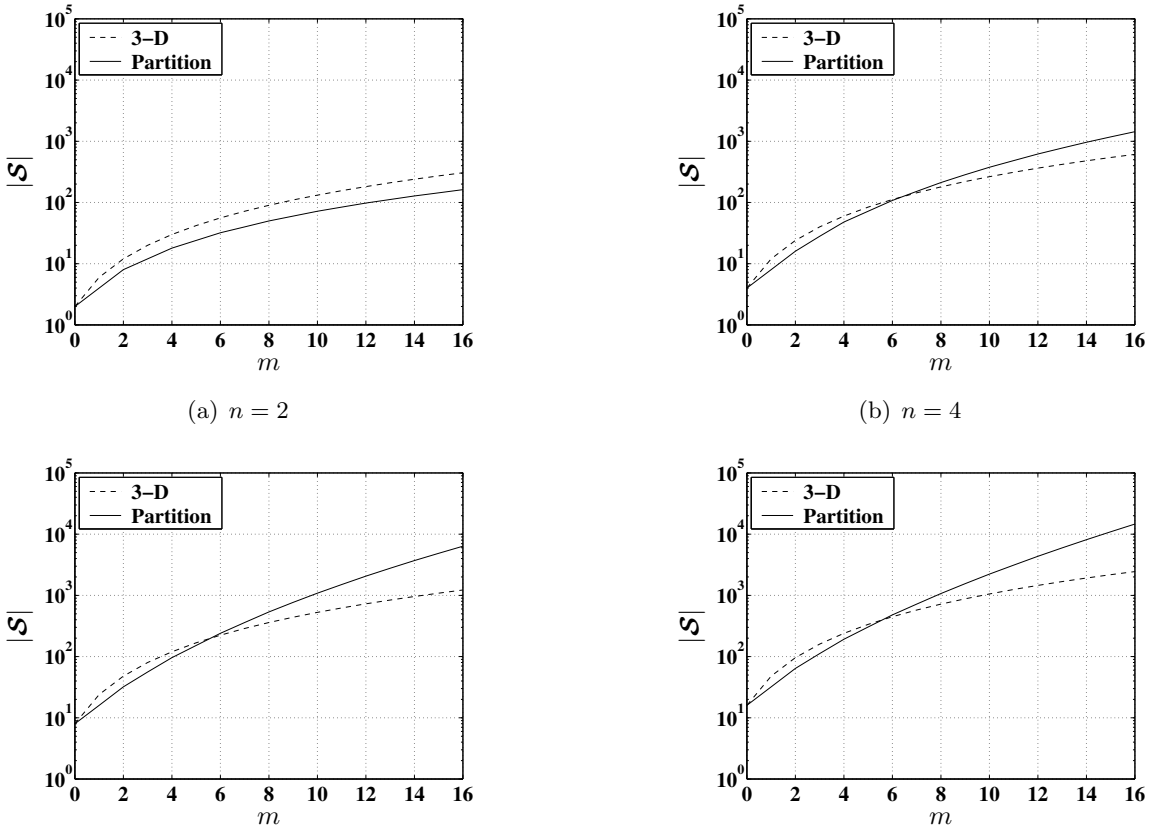
[6] These larger switches are often better handled by approximate models which become more accurate with increasing switch size because of the lower correlations in the addresses of the buffered packets.

$$|\boldsymbol{\mathcal{S}}| = (h+1)\frac{(m+2)(m+1)}{2} = (h+1)\left\langle \begin{matrix} m+1 \\ 2 \end{matrix} \right\rangle. \tag{92}$$

For the partition model, $|\boldsymbol{\mathcal{S}}|$ is given by

$$|\boldsymbol{\mathcal{S}}| = (I+1)\sum_{b=0}^{m} |\boldsymbol{\Psi}(b,g)|, \tag{93}$$

where $I = h$, $c = 1$ for comparison with the 3-D model which does not support channel-grouped switches. Figure 22 shows the required state space for switches of typical sizes under bursty traffic. Surprisingly, the exact partition model actually requires *fewer* states than the approximate 3-D model for $n = 2$ switches and for $n = 4, 8, 16$ switches with $m < 6$. Although the 3-D model does scale better for larger switches and buffers, it does introduce some "redundant" states for the smaller ones.



(a) $n = 2$

(b) $n = 4$

(c) $n = 8$

(d) $n = 16$

Figure 22: Number of system states required for the model of this work ("partition" model) compared to that of Pattavina's "3-D" model [24].

The required $|\boldsymbol{\mathcal{S}}|$ for the partition model is within an order of magnitude of that of the 3-D model, even for switches as large as $m = n = 16$.

4.8 Numerical Results

The two-state source has two parameters more than the random source does. As a result, switch systems under bursty traffic have an even larger parameter space than those under random traffic. A simple TS source can be described in terms of (p, p_{on}, p_{off}) or in an alternative basis such as (β, p, σ) or (β, λ, σ). The resultant traffic properties are more easily deduced from these latter two forms. The ETS source has the additional parameters p_i instead of just the single value p. However, p_{on} and p_{off} are determined for a given set of p_i and any two of β, λ, or σ. Of course, any $p_j = 1$ implies that the others, p_k ($k \neq j$), are zero because $\sum_{i=0}^{c} p_i = 1$.

Some numerical analytic and simulation (SMELTER and SONSIM) results for OSMA switches under bursty traffic will now be presented. Figure 23 shows $\mathcal{P}_{\text{loss}}$ for $n = 8$, $c = 1, 4$, $0 \leq m \leq 8$ switches driven by FOB ETS sources ($\lambda = 8$, $p_c = 1.0$, $\sigma = 0.3$) with $\beta \approx 2.9$. The simulation results validate those from the exact analytic model. The FOB source is very demanding upon a switch as it entirely fills the link

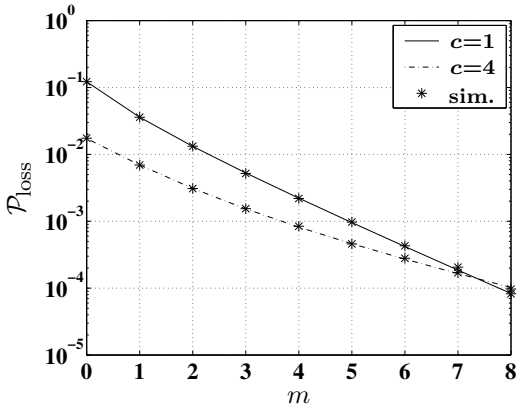


Figure 23: $\mathcal{P}_{\text{loss}}$ for $n = 8$ switches under ETS traffic, $\lambda = 8$, $p_c = 1.0$, $\sigma = 0.3$, ($\beta \approx 2.9$ and $\beta_S = 1, 2.533$ for $c = 1, 4$, respectively).

in every timeslot that it is on. Furthermore, the average burst length of eight is somewhat long. It is not surprising that the switch exhibits such high loss rates when subjected to these traffic conditions. The beneficial $\mathcal{P}_{\text{loss}}$ effect of channel grouping on the outputs is almost completely nullified by the very large number of simultaneously arriving packets in the multichannel input links. In fact, for $m = 8$, the $c = 4$ switch actually has a slightly higher loss rate than its lower bandwidth counterpart.

Figure 24 shows the same $c = 1$ switch under less bursty traffic ($\lambda = 8$, $p = 0.6$, $\sigma = 0.3$, yielding $\beta = 1.75$) and under random traffic, with loads of 0.3 and 0.6, for comparison. Successive packet arrivals tend to result in an increase in the number of packets in the buffer. For this reason, a buffered switch under bursty traffic has a $\mathcal{P}_{\text{loss}}$ at least as large as that of the same switch under random traffic of the same load. However, $\mathcal{P}_{\text{loss}}$ under bursty traffic will not be greater than that under random traffic with load σ_{on} — the off periods of the bursty sources reduce output port contention and allow the buffer a chance to clear. This intuitive line of reasoning provides a means to establish a floor and a ceiling to $\mathcal{P}_{\text{loss}}$ under bursty traffic, in terms of random traffic with loads of σ and σ_{on}, respectively (Figure 24). Unfortunately, this range can span several orders of magnitude and, therefore, provides a poor substitute

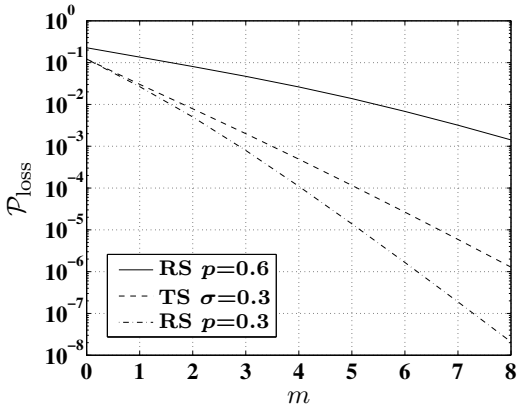


Figure 24: $\mathcal{P}_{\text{loss}}$ for $n = 8$, $c = 1$ switches under random (RS) and bursty (TS) traffic.

to having the "real" bursty analytic results shown in the middle curve. However, the observation may be useful as a rule-of-thumb or to serve as a base upon which to interpolate when a more general model, like the one of this chapter, is unavailable or too computationally demanding under bursty traffic.

4.8.1 The Effect of Burstiness on the Packet Loss Rate

Figure 25 shows how β affects $\mathcal{P}_{\text{loss}}$ for the same ($n = m = 8$, $c = 1$) switch as in Figure 24. For these traffic parameters, the valid range of β is $0 < \beta < 2$. The limits are not inclusive because the end points result in invalid Markov models. The four order-of-magnitude range that $\mathcal{P}_{\text{loss}}$ spans clearly shows that variations in β can have a profound effect on $\mathcal{P}_{\text{loss}}$. For bursty traffic ($\beta > 1$), $\mathcal{P}_{\text{loss}}$ stays between its floor and ceiling (as previously discussed). For $\beta = 1$, the two-state source behaves exactly like a random source and the model agrees perfectly with that of the simpler random case. It was pointed out in Section 4.1.2 that the two-state source can generate antibursty ($\beta < 1$) traffic. When $\beta < 1$, $\mathcal{P}_{\text{loss}}$ is actually lower than that under random traffic. The source "backs off" after sending a packet so it tends not to send two or more packets consecutively. This gives the buffer more of a chance to clear between arrivals than

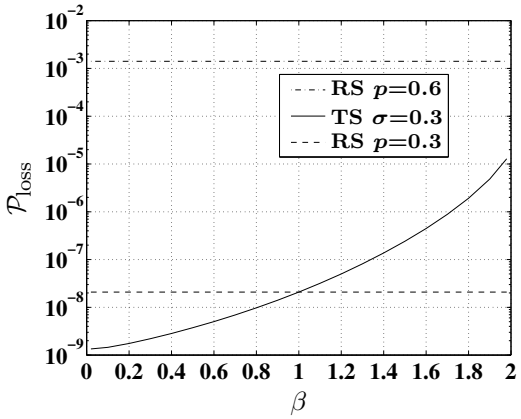


Figure 25: Effect of β on $\mathcal{P}_{\text{loss}}$ for an $n = 8$, $c = 1$, $m = 8$ switch. Two-state source (TS) parameters are β, $p = 0.6$, $\sigma = 0.3$. $\mathcal{P}_{\text{loss}}$ under $p = 0.3,\ 0.6$ random traffic (RS) is also shown for comparison.

with the random source, which will send packets out with the same 30% probability regardless of what it did in the previous timeslot.

Figure 26 shows that $\mathcal{P}_{\text{loss}}$ can be very sensitive to changes in spatial burstiness as well. The traffic source used for Figure 26 is an ERS (memoryless) source with a binomial distribution for p_i at $\beta_S = 1$. For $\beta_S > 1$, p_4 was incremented to fixed values at the proportional expense of p_{1-3} as shown for selected data points in Table 1. Because there are more degrees of freedom with the p_i parameters (even with the σ and $\sum_{i=0}^{c} p_i = 1$ constraints for $c > 2$) than with the single parameter β_S, there are many different valid p_i sets, each of which may result in a different $\mathcal{P}_{\text{loss}}$ for a given switch, and many of which may have the same β_S. Indeed, as can easily be seen from Equation 75, β_S is constant for ERS sources if p_0 and σ are held constant regardless of the values of p_i $(i > 0)$. Nevertheless, β_S can be a very helpful first order rule of thumb in predicting the effect on $\mathcal{P}_{\text{loss}}$ a given change in p_i will have.

Notice in Figure 26, for the region of small β_S, that $\mathcal{P}_{\text{loss}}$ is very sensitive to changes in β_S and it becomes less so with increasing β_S. This effect may be due, at least in part, to the fact that p_4 initially increases rapidly, as measured as a percentage of p_4, and then increases more slowly for larger values of p_4 and corresponding β_S (see

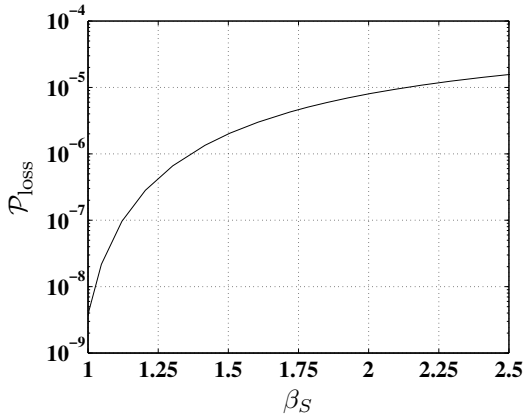


Figure 26: Effect of β_S on $\mathcal{P}_{\text{loss}}$ for an $n = 4$, $c = 4$, $m = 8$ switch under ERS traffic with $\sigma = 0.3$.

the p_4 column of Table 1). Shifts in the occupancy distribution of traffic such as this example have some importance in the analysis of networks, so this topic will be revisited in Chapter 5.

Table 1: Selected data points for the traffic parameters used in Figure 26.

p_0	p_1	p_2	p_3	p_4	β_S
0.240	0.412	0.265	0.076	0.008	1*
0.385	0.282	0.181	0.052	0.1	1.235
0.542	0.141	0.091	0.026	0.2	1.661
0.7	0	0	0	0.3	2.533

*binomial distribution

4.8.2 The Effect of Load on the Packet Loss Rate

As with random traffic, a change in the load typically has a major effect on the loss rate as shown in Figure 27. Notice the similarity between Figure 27 and Figure 18. However, keeping λ constant while varying σ results in changes in β. For example, in Figure 27, the traffic has $\beta = 7.5$ at $\sigma = 0.1$, but has $\beta = 0.94$ (antibursty) at

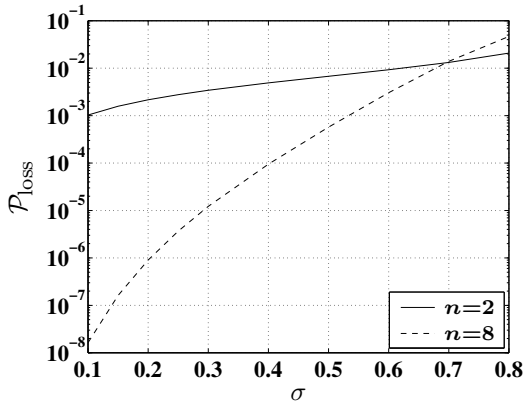


Figure 27: Effect of σ on $\mathcal{P}_{\text{loss}}$ for $n = 2, 8$, $c = 1$, $m = n$ switches under bursty traffic of $\lambda = 4$, $p = 1.0$, $(0.9 < \beta \leq 7.5)$.

$\sigma = 0.8$.[7] The increase of β with decreasing σ partially offsets the reduction in $\mathcal{P}_{\text{loss}}$ afforded by the lighter load. Thus, the loss rates of Figure 27 are several orders of magnitude greater than those of Figure 18 for the lighter loads.

If instead we choose to hold β constant and allow p to vary with σ, the result is shown in Figure 28. Because $\beta = 1.5$ is a relatively mild degree of burstiness, the loss rates are much closer to those of Figure 18 — and this is true in spite of the fact that p reaches unity at $\sigma = 0.5$.[8]

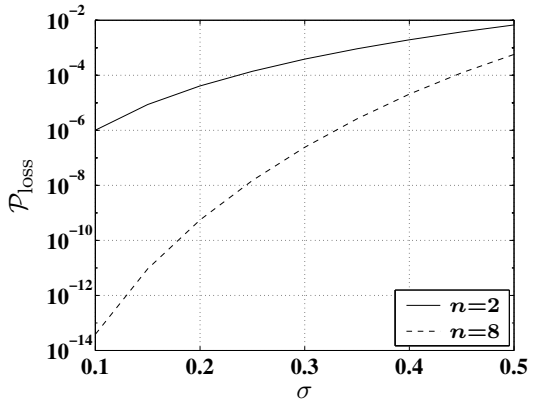


Figure 28: Effect of σ on $\mathcal{P}_{\text{loss}}$ for $n = 2, 8$, $c = 1$, $m = n$ switches under bursty traffic of $\beta = 1.5$, $\lambda = 4$, $(0.2 \leq p \leq 1.0)$.

4.8.3 The Effect of Burst Length on the Packet Loss Rate

Perhaps surprisingly, $\mathcal{P}_{\text{loss}}$ can be relatively insensitive to changes in λ if β, β_S, and σ are held constant as shown in Figure 29. By itself, an increase in λ will result in a non-negligible increase in $\mathcal{P}_{\text{loss}}$ because longer sustained bursts have a better chance of overflowing the buffer. However, holding β and σ constant requires that p be decreased as λ is increased. For example, in Figure 29 at $\lambda = 2$, $p = 0.9$, but at

[7]For the parameters used in Figure 27, σ cannot exceed 0.8 because this would require p_{on} to be greater than unity.

[8]For these parameters, $\sigma > 0.5$ is not possible as this would require $p > 1$.

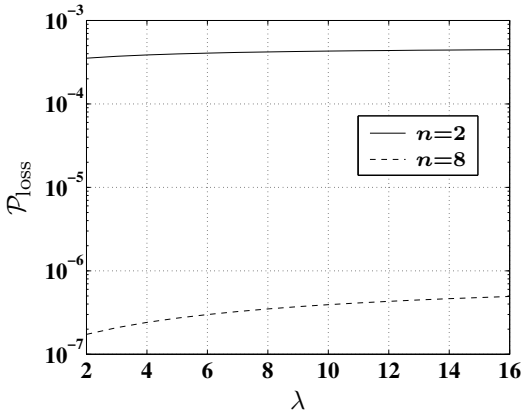


Figure 29: Effect of λ on $\mathcal{P}_{\text{loss}}$ for $n = 2, 8$, $m = n$ switches under bursty traffic of $\beta = 1.5$, $\sigma = 0.3$, $0.47 < p \leq 0.9$.

$\lambda = 16$, $p \approx 0.47$. This decrease in the intensity of the bursts offsets the effect of the longer burst length. Therefore, λ or p alone can be weak affecters of $\mathcal{P}_{\text{loss}}$.

4.8.4 Reducing the Packet Loss Rate via Channel Grouping and/or Increasing the Buffer Size

Judging the effects of channel grouping on $\mathcal{P}_{\text{loss}}$ under bursty traffic is somewhat involved because of the different ways a bursty traffic model can be interfaced to a multichannel link. One approach is to speed up a source so as to reduce λ by a factor of c. With such an arrangement, some degree of temporal burstiness is "exchanged" for the presence of spatial burstiness. Figures 30 and 31 show the results of this approach for $n = 2$ and $n = 4$ switches, respectively, under bursty traffic of $\lambda = \frac{8}{c}$, $p_c = 1$ and $\sigma = 0.5$. The loss rate improvement with channel grouping is quite modest because: (1) Reductions in temporal burstiness ($\beta = 1.75, 1.5, 1$) are offset by corresponding increases in spatial burstiness ($\beta_S = 1, 1.5, 1.875$) for respective increases in the channel grouping factor ($c = 1, 2, 4$); (2) In all cases, the load remains constant ($\sigma = 0.5$). This second issue raises the same concern brought up in Section 3.5 (Figure 13). Namely, is it fair to use the extra bandwidth or should the load be reduced proportionally with increases in c so as to keep the total switched

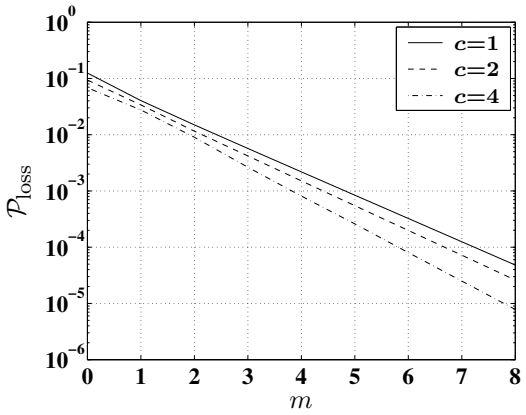


Figure 30: $\mathcal{P}_{\text{loss}}$ for $n = 2$, $c = 1, 2, 4$ switches under bursty traffic, $\lambda = \frac{8}{c}$, $p_c = 1$, $\sigma = 0.5$.

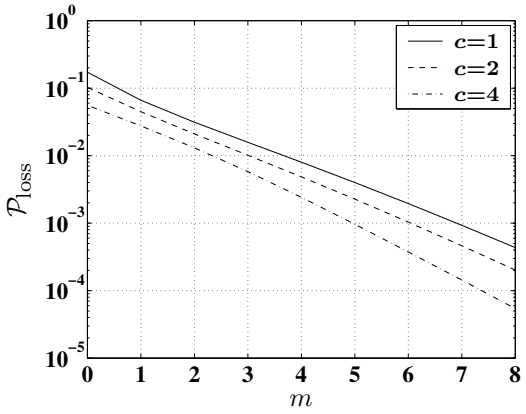


Figure 31: $\mathcal{P}_{\text{loss}}$ for $n = 4$, $c = 1, 2, 4$ switches under bursty traffic, $\lambda = \frac{8}{c}$, $p_c = 1$, $\sigma = 0.5$.

bandwidth constant? Figures 32 and 33 show that channel grouping is more effective in reducing $\mathcal{P}_{\text{loss}}$ when the additional link bandwidth is not exploited.

Channel grouping is far more effective when applied only to the outputs (Figure 34). Not shown in Figure 34 is the $n = 2$, $r = 2, 4$ and $n = 4$, $r = 4$ cases which are completely lossless because output link contention is impossible. However, as mentioned in Section 3.5, such switches may not be practical for exclusive use in networks because of fanout growth.

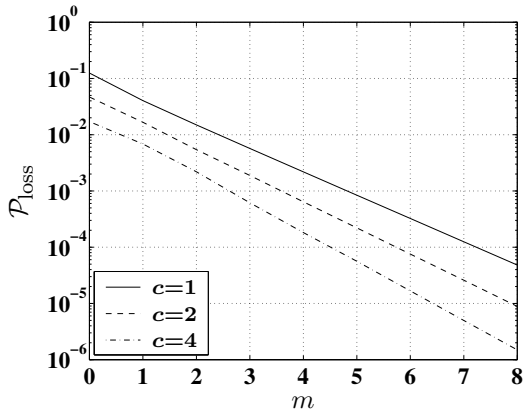


Figure 32: $\mathcal{P}_{\text{loss}}$ for $n = 2$, $c = 1, 2, 4$ switches under bursty traffic, $\lambda = \frac{8}{c}$, $p_c = 1$, $\sigma = \frac{0.5}{c}$.

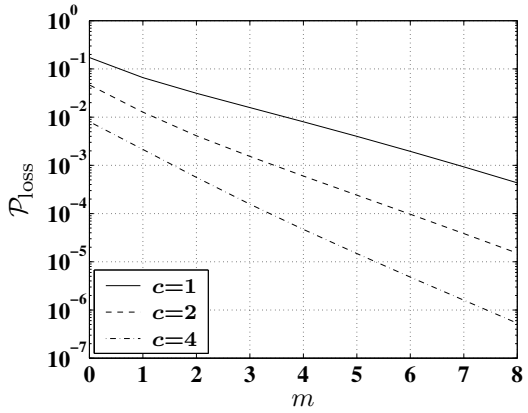


Figure 33: $\mathcal{P}_{\text{loss}}$ for $n = 4$, $c = 1, 2, 4$ switches under bursty traffic, $\lambda = \frac{8}{c}$, $p_c = 1$, $\sigma = \frac{0.5}{c}$.

Finally, instead of reducing the load and/or using channel grouping, $\mathcal{P}_{\text{loss}}$ can be lowered via increasing (perhaps dramatically) the size of the buffer (Figure 35). For example, comparing Figure 35 with Figure 33, it can be seen that, for the $n = 4$, $c = 1$, $m = 8$ switch under the given traffic conditions, if a loss rate of 10^{-6} or lower is needed, increasing c to four (and not exploiting the extra bandwidth) is equivalent (as far as $\mathcal{P}_{\text{loss}}$ is concerned) to increasing the buffer size to 16. Which approach is best depends on the cost associated with implementing each. Because costs typically vary

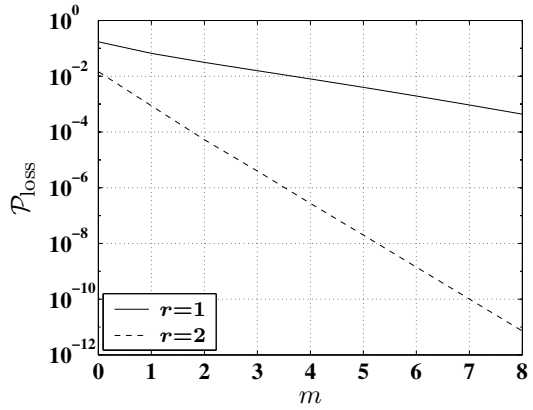


Figure 34: $\mathcal{P}_{\text{loss}}$ for $n = 4, s = 1, r = 1, 2$ switches under bursty traffic, $\lambda = 8$, $p = 1$, $\sigma = 0.5$.

nonlinearly with respect to the various switch parameters, determining the optimal solution can be a nontrivial matter. In any event, it is obvious that meeting a given $\mathcal{P}_{\text{loss}}$ goal becomes much more difficult (and expensive) to achieve as the burstiness of the traffic increases.

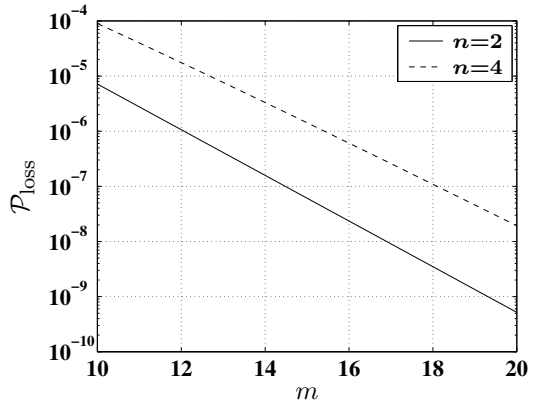


Figure 35: $\mathcal{P}_{\text{loss}}$ for $n = 2, 4$, $c = 1$ switches with large buffers under bursty traffic, $\lambda = 8$, $p = 1.0$, $\sigma = 0.5$, (yielding $\beta = 1.75$).

Chapter 5

Networks of OSMA Switches

Switches are the building blocks of networks. Because switches are usually used within the context of a network, it is important to be able to predict their performance when they are interconnected. The historical dependencies of a network are contained in the states of *all* of its switches. Therefore, an exact Markov model of the entire network requires a state space cardinality equal to the product of the cardinalities of the individual switch state spaces. Unfortunately, this makes an exact Markov network model impractical for all but the most trivial networks, even with the heavy switch state space reduction employed in this work. Previous works [24], [55], [56], [58] that analyzed networks of ESMP switches utilized approximate models of the switches and/or made unrealistic assumptions about the properties of the interstage traffic. (The predicted overall network loss rates from some of these models can deviate from the true values by several orders of magnitude, even at relatively high (10^{-2}) loss rates [24], [56]). Furthermore, none of these models encompass networks that utilize multichannel links.

5.1 Interstage Traffic Approximation

The approach used in this work is to combine the exactness of the model for the internal operation of the switch, as developed in the previous chapters, with a "reasonably close" approximation of the interstage traffic. In this method, a multistage network is analyzed through the use of "virtual sources" which approximate the interstage traffic (Figure 36). Hence, an analysis is carried out for each stage and then the results are used to determine the parameters of the virtual sources that feed the next stage. Successive stages are analyzed in this manner until the end of the network is reached.

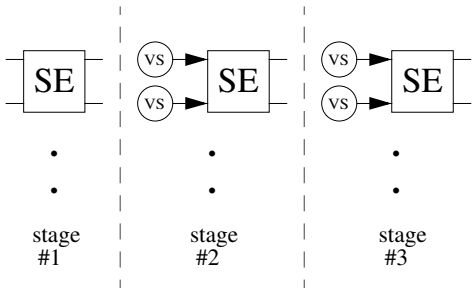


Figure 36: Approximating interstage traffic with virtual sources.

There are constraints on the types of networks that can be analyzed with this approach. Throughout this work, it was required that the input links (sources) to a switch be independent from one another (i.e., there are no inter-link traffic correlations) and that they must have the same parameters. Furthermore, the destination addresses of the packets must be completely random. Therefore, the networks to be analyzed are subject to all of the following constraints:

1. Different input links to a switch must carry traffic with the same parameters — the upstream structures must look identical for all input links.

2. There must not be any inter-input-link traffic correlations — different input links to a switch must not have paths leading to a common upstream switch or source.

3. The routing actions of an upstream switch must not introduce correlations in the packet destination addresses to downstream switches. This will be true if switches in different stages self-route on different address fields.

One very broad category of network that can have these properties (loosely referred to here as a "cascade") is shown in Figure 37. Perhaps surprisingly, Banyan networks can also satisfy these requirements as the different inputs to each switch do not have paths to a common upstream switch. Because the switches in each stage "see" traffic with the same properties, one needs to calculate the performance of only a single switch in a stage to know the performance of every switch in that stage.

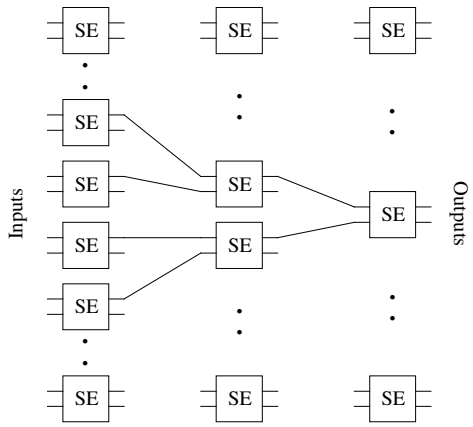


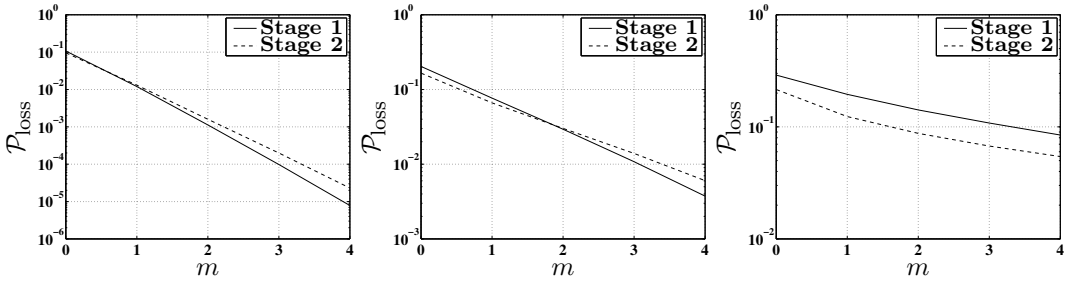
Figure 37: A cascade of switches.

5.2 Output Traffic of OSMA Switches

The action of the switch fabric and buffer significantly alters the traffic that passes through a switch. First of all, the switch combines input packets with the same destination addresses and then sends them to the appropriate output links. Thus, the traffic of an output link is a type of selective composite of the traffic in all the input links. Secondly, any dropped packets serve to reduce the total output traffic load. Finally, the action of the buffer serves to store any contending packets and defer their output until the next available timeslot(s). Therefore, the output traffic of a switch, and hence, the interstage traffic of a network, tends to be somewhat bursty — even if the input traffic is purely random.

5.2.1 Interstage Load and Temporal Burstiness

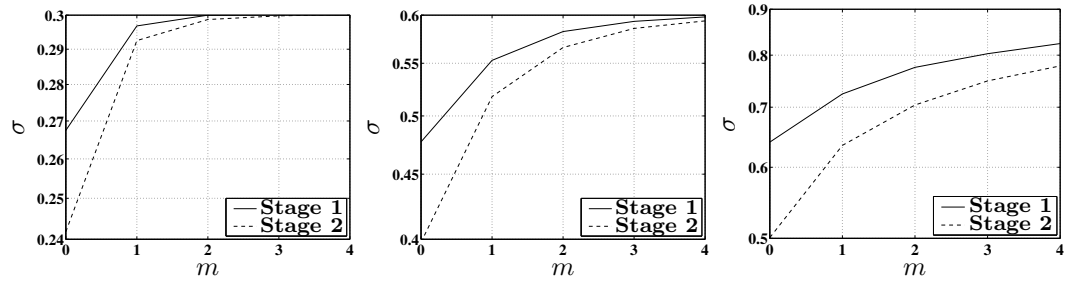
A simulation (via SONSIM) was carried out on a two-stage 16×16 Banyan network composed of 4×4 switches ($c = 1,\ 0 \leq m \leq 4,\ n = 4$) under random traffic loads of 0.30, 0.60 and 0.90. $\mathcal{P}_{\text{loss}}$ for switches in each stage is shown in Figure 38. Most interesting is the "crossover point," easily seen in Figure 38(b), where the loss rates in each stage are equal. For the 30% load, the crossover point occurs for very small m. As the load increases, it takes a larger m to make the loss rates equal and the



(a) RS input load = 30%. (b) RS input load = 60%. (c) RS input load = 90%.

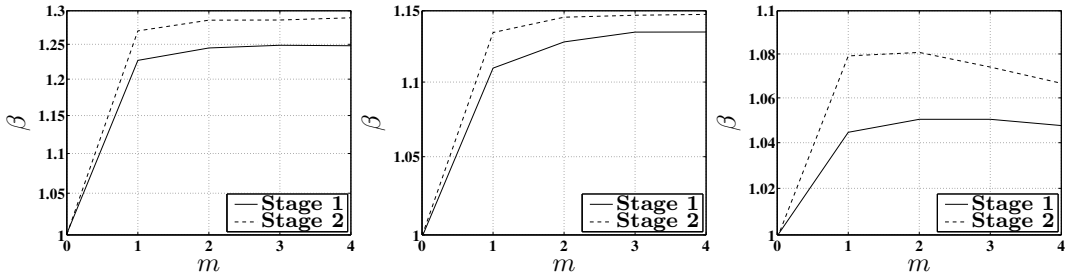
Figure 38: $\mathcal{P}_{\text{loss}}$ from simulation for a 16×16 Banyan network of 4×4 switches with $c = 1,\ 0 \leq m \leq 4$ under random traffic.

crossover point moves rightward. At the 90% load, the crossover point cannot be seen in Figure 38(c) because it is to the right of the graph. An explanation for the varying $\mathcal{P}_{\text{loss}}$ between the stages can be seen in Figures 39 and 40, which show the measured (via simulation) average load (σ) and temporal burstiness factor (β), respectively, at the *output* of each stage. Increasing m increases both σ and β of a switch's output traffic. The buffer acts to resolve contention by temporally shifting contending output packets into the next available timeslot(s), thereby creating a "burst" that lasts until the switch is able to clear itself of packets with the given destination address. For small m and/or large input loads, the loss rate of the first stage is high enough to significantly decrease the traffic load seen by stage 2. As a result, stage 2 exhibits a



(a) RS input load = 30%. (b) RS input load = 60%. (c) RS input load = 90%.

Figure 39: σ at the output of each stage as measured from simulation.



(a) RS input load = 30%. (b) RS input load = 60%. (c) RS input load = 90%.

Figure 40: β at the output of each stage as measured from simulation.

lower loss rate than stage 1. However, as m increases, stage 1 looses fewer packets and the burstiness of its output traffic increases. Therefore, the additional buffer cells do not decrease the $\mathcal{P}_{\text{loss}}$ of stage 2 as fast as they do in stage 1 — resulting in the inevitable intersection of the $\mathcal{P}_{\text{loss}}$ curves.

There are other interesting observations from Figure 40. For $m = 0$, $\beta = 1$ at the outputs for all input loads. Without buffers, the switch becomes memoryless and the output traffic is random (albeit with a smaller load than the input traffic). Note the rapid increase in β as m is initially increased beyond zero. Later additions to m do not increase β as much. β increases are the direct result of the additional buffer cells being used. As m increases beyond a certain point, these additional cells are used so infrequently that they do not noticeably affect the output traffic. In this way, Figure 40 is closely related to Figure 14. The slight decrease in β, seen for the 90% input load case as m increases beyond two, is interesting as well. As mentioned in Section 4.1.2, β cannot be made as large as σ increases.[1] So, under high loads as m increases, the loads at the outputs of the switches increase to the point that the "ceiling" on β decreases and β is forced to become slightly lower. The same effect can be seen in a different way; for a given m, β decreases with increasing input load as can be seen by comparing Figures 40(a,b,c).

[1]For an extreme example, at $\sigma = 1$, β *must* be equal to one. The source/switch cannot burst because every packet slot is always occupied.

5.2.2 Interstage Spatial Burstiness

To explore the evolution of interstage spatial burstiness independently of β, a simulation (via SONSIM) was carried out on a four-stage 16×16 Banyan network composed of 2×2 bufferless switches ($c = 4,\ m = 0,\ n = 2$) under a random traffic (each channel driven independently, $\sigma = p$) load of 0.30.

In this network, temporal burstiness cannot occur as every switch and input source is memoryless. Thus, $\beta = 1$ at every link in the network. The measured interstage traffic properties as well as $\mathcal{P}_{\text{loss}}$ for each stage is shown in Table 2. As expected,

Table 2: Interstage traffic properties at the output of each stage for a 16×16 Banyan network of 2×2 bufferless switches, $c = 4$, under a random traffic load of 0.30.

Stage #	Link Occupancy Probabilities %					σ	β_S	$\mathcal{P}_{\text{loss}}$
	0	1	2	3	4			
Input Source	24.05	41.08	26.44	7.61	0.81	0.300	1.001	n/a
Stage 1	27.31	38.42	23.76	8.38	2.14	0.299	1.044	2.54×10^{-3}
Stage 2	28.87	37.16	22.70	8.58	2.69	0.298	1.064	4.39×10^{-3}
Stage 3	29.52	36.73	22.25	8.60	2.91	0.297	1.072	5.20×10^{-3}
Stage 4	29.93	36.57	22.00	8.57	2.93	0.295	1.075	5.34×10^{-3}

the input source link occupancy probabilities (q_i) have a binomial distribution which is not spatially bursty. However, as one progresses through the network, the occupancy distribution shifts to favor the higher occupancy levels. For example, the probability of having four packets in a link at an input source is 0.81%, but this increases by almost a factor of four to 2.91% at the output of stage 3 (which is the input to stage 4). This change in the occupancy distribution causes the later stages to see traffic of increasing spatial burstiness. At the input of stage 4, β_S has increased to 1.072, which causes stage 4 to experience more than twice the loss rate of stage 1 — even though stage 4 carries a slightly lighter traffic load because of the losses in the earlier stages.

Bufferless switches under random traffic are simple enough systems to allow for some direct analysis of this effect. The link occupancy probabilities for an input to a switch driven by a random source is given by

$$\mathcal{P}_{\text{in},i} = \binom{s}{i} p^i (1-p)^{s-i}. \tag{94}$$

However, the probability of i packets appearing in a given tagged output link of the switch, $\mathcal{P}_{\text{out},i}$, is given by

$$\mathcal{P}_{\text{out},i} = \begin{cases} \binom{hs}{i} (\frac{p}{g})^i (1-\frac{p}{g})^{hs-i} & : \quad 0 \le i < r \\ \sum\limits_{j=r}^{hs} \left(\binom{hs}{j} (\frac{p}{g})^j (1-\frac{p}{g})^{hs-j} \right) & : \quad i = r \end{cases}. \tag{95}$$

The summation in the equation for $i = r$ is needed to count the cases when output link contention occurs. For symmetric switches ($h = g = n,\ s = r = c$), the ratio of the probabilities of full occupancy of the output and input links is given by

$$\frac{\mathcal{P}_{\text{out},c}}{\mathcal{P}_{\text{in},c}} = \frac{\sum\limits_{i=c}^{nc} \left(\binom{nc}{i} (\frac{p}{n})^i (1-\frac{p}{n})^{nc-i} \right)}{p^c}, \tag{96}$$

which is plotted as a function of n and c for $\sigma = p = 0.5$ in Figure 41. It is surprising

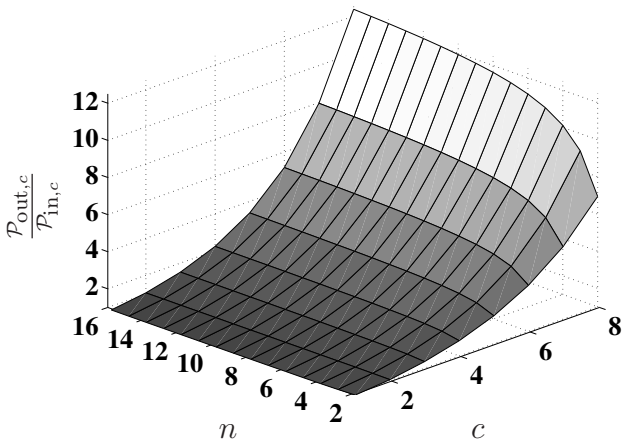


Figure 41: $\frac{\mathcal{P}_{\text{out},c}}{\mathcal{P}_{\text{in},c}}$ versus n and c for bufferless switches under a random traffic load of 0.5.

to learn that the larger switches can produce traffic that completely fills the output links more than an order of magnitude more often than the random, binomial, input traffic.

5.2.3 Eigentraffic

This is not to say that traffic always becomes increasingly spatially or temporally bursty as it progresses through the stages of a network. Because, in this work, the packets of a (spatial or temporal) burst always have uncorrelated addresses, they are always dispersed among all of the output links when passing through a switch. This effect limits the maximum burstiness traffic can sustain as it travels through switches. Thus, switches with a given set of parameters under a given traffic load favor a "natural" level of temporal and spatial burstiness and tend to transform input traffic properties toward these "eigenvalues."

5.3 Determining the Virtual Source Model

The traffic at the output of a switch is complex. It is the result of the actions of a system that has hundreds, if not thousands, of states. If the switch is under bursty traffic, the additional input source states make the resulting output traffic even more difficult to accurately model. The most general source model developed in this work is the ETS source. By an appropriate choice of parameters, an ETS source can serve as any RS, ERS, or TS source. It is fully capable of supplying a multichannel link with traffic with an arbitrary packet occupancy probability distribution. This flexibility is obtained with a mere two states. In this work, ETS sources will serve as the interstage link virtual sources. In fact, the primary reason for extending the switch model to include ETS inputs is to facilitate the use of the ETS source as an approximate interstage traffic model.

The process of matching ETS parameters to switch output traffic will now be developed.

5.3.1 Output Traffic Information Required from the Switch Model

Additional information from the model is needed to ascertain the spatial and temporal properties of the switch's output traffic. The output link occupancy distribution is obtained via $\mathcal{E}_{\text{links},i|x}$ which is the expected number of output links that have i packets given that the switch/source system is in state number x. $\mathcal{E}_{\text{links},i|x}$ is generated at the same time as $\mathbf{P}$, $\mathcal{E}_{l|x}$ and $\mathcal{E}_{w|x}$ (see Section 4.6). In the same manner as $\mathcal{E}_{l|x}$ and $\mathcal{E}_{w|x}$, $\mathcal{E}_{\text{links},i|x}$ is constructed by successive additions within the loops of the model algorithm:

$$\mathcal{E}'_{\text{links},i|x} = \mathcal{E}_{\text{links},i|x} + J_i \mathcal{P}_{\mathcal{A}} \mathcal{P}_L, \tag{97}$$

where J_i is the number of output links that have i ($0 \leq i \leq r$) packets (i.e., the number of O'_k ($0 \leq k < g$) that are equal to i).

Ascertaining the temporal properties of the output traffic and matching an ETS source to it is complicated by two issues: (1) Knowing q_i and β of the traffic in a link does not provide one with enough information to solve for p_i, p_{on} and p_{off}; and (2) the Markov nature of the model limits the memory of the system to that which is contained in the current state. Obtaining information beyond this (say, to track the traffic of a particular output) would require expanding the system states — which is highly undesirable.

The first difficulty arises because, from an external point of view, it is impossible to know exactly when a burst ends while monitoring the ETS output traffic, if and only if, the ETS source is allowed not to emit packets when on ($p_0 \neq 0$). Defining output bursts to consist only of timeslots in which a given output link has at least one packet has a physical basis in the switch system. Namely, when the buffer and input links become devoid of packets that are addressed to the given output link, the switch system effectively becomes memoryless with (and only with) regard to that output link, and hence, this event represents the end of the "output burst." This is not to

suggest that the burst length of the input sources is not taken into consideration, but merely that it is done so through its action on the buffer. Also, even if the model's complexity were to be increased so as to relax the $p_0 = 0$ restriction, and to increase λ of the ETS source by allowing bursts to contain "empty" timeslots, the effect on predicting switch $\mathcal{P}_{\text{loss}}$ performance would be minimal because variations of λ with fixed σ, q_i and β typically have very little impact on $\mathcal{P}_{\text{loss}}$ as was discussed in Section 4.8.3. Furthermore, the allowed upper range of p_0, and consequently λ, can be severely limited by the other constraints (more so than the example of Figure 29). For all of the above reasons, $p_0 = 0$ will be required for the ETS virtual source match.

The second difficulty can be overcome by realizing that, in any given system state, one can easily "look forward" by one timeslot — which is already required during the course of evaluating the state transitions. This simple transition based analysis is obviously sufficient to determine β as can easily be seen from its "$\mathsf{packets}\longrightarrow\mathsf{packets}$"[2] based definition in Equation 69. Note that knowing the $\mathsf{packets}\longrightarrow\mathsf{packets}$ probability implies knowing the "$\mathsf{packets}\longrightarrow\mathsf{empty}$"[3] probability because they both have to sum to unity. Furthermore, either of these two in conjunction with q_i determines the $\mathsf{empty}\longrightarrow\mathsf{empty}$ and $\mathsf{empty}\longrightarrow\mathsf{packets}$ probabilities in the same way that the q_i can be viewed as steady-state probabilities of a two-state Markov chain with the "$\mathsf{X}\longrightarrow\mathsf{Y}$" serving as transition probabilities. Knowing the $\boldsymbol{\pi}$ (q_i) and *any* single transition probability $(\mathsf{X}\longrightarrow\mathsf{Y})$ determines the entire $\mathbf{P}$ for a two-state Markov chain. This relationship between the $\mathsf{X}\longrightarrow\mathsf{Y}$ properties of the traffic will hold even if the traffic was produced by a more complex Markov chain[4] because we are concerned only with observed average traffic properties (which from their definitions have a simple Markovian relationship) and not the underlying mechanics which actually gave rise to the traffic. The above discussion is important only because, in practice, it is easier to obtain (and then match) the

[2]The probability that a link with one or more packets will have at least one packet in the next timeslot.
[3]The probability that a link with one or more packets will not have any packets in the next timeslot.
[4]Or even if it is the result of a "real world" process of unknown mechanism.

$\mathsf{empty}\longrightarrow\mathsf{packets}$ transition probability directly from the switch model (as will be done in Equation 102) than the $\mathsf{packets}\longrightarrow\mathsf{X}$ probabilities as the former, unlike the latter, does not involve the switch buffer's contents. With either approach, all of the $\mathsf{X}\longrightarrow\mathsf{Y}$ probabilities, as well as β, are guaranteed to be matched.

Finally, because the addresses of the packets are required to be uniform and random, the traffic properties of a given tagged output can be ascertained from the overall output traffic of the switch ($\mathcal{E}_{\text{links},i|x}$) without having to resort to additional system states to track the traffic of a specific output.

5.3.2 Matching Obtained Traffic Parameters to an ETS Source

It is trivial to obtain $\mathcal{P}_{\text{out},i|x}$, the probability of a given output link having i packets, given that the switch is in system state number x, from $\mathcal{E}_{\text{links},i|x}$:

$$\mathcal{P}_{\text{out},i|x} = \frac{\mathcal{E}_{\text{links},i|x}}{g}. \tag{98}$$

Using $\mathcal{P}_{\text{out},i|x}$ and the law of total probability, one can obtain $\mathcal{P}_{\text{out},i}$, which is the probability of a given output link having i packets for the overall system:

$$\mathcal{P}_{\text{out},i} = \sum_{\boldsymbol{\mathcal{S}}} \pi_x \mathcal{P}_{\text{out},i|x} = \sum_{\boldsymbol{\mathcal{S}}} \frac{\pi_x \mathcal{E}_{\text{links},i|x}}{g}. \tag{99}$$

$\mathcal{P}_{\text{out},i}$ is the same as q_i for the traffic of any given output link, so the spatial part of the description of the output traffic is now complete.

For the temporal aspects of the traffic, two preliminary results are needed. $\mathcal{P}_{x|\text{out},i}$, the probability of being in system state number x given that a tagged output link has i packets (i.e., the "inverse" of Equation 98) can be obtained via Bayes' rule:

$$\mathcal{P}_{x|\text{out},i} = \frac{\pi_x \mathcal{P}_{\text{out},i|x}}{\mathcal{P}_{\text{out},i}}, \tag{100}$$

of which we will be mainly interested in the $i = 0$ case. $\mathcal{P}_{\text{srcout}|x_{\mathcal{I}}}$, the probability that the input sources will emit at least one packet addressed to a tagged output link, given that the input sources are in input state number $x_{\mathcal{I}}$, is given by:

$$\mathcal{P}_{\text{srcout}|x_{\mathcal{I}}} = \sum_{\alpha=1}^{hs} \left[\mathcal{P}_{\alpha|x_{\mathcal{I}}} \left(1 - \left(\frac{g-1}{g} \right)^{\alpha} \right) \right], \tag{101}$$

where $\mathcal{P}_{\alpha|x_{\mathcal{I}}}$ is obtained using the appropriate equation for $\mathcal{P}_{\alpha}$ from Section 4.4 with $\mathcal{S}_x^{\mathcal{I}} = \mathcal{I}_{x_{\mathcal{I}}}$.

Now, using $\mathcal{P}_{x|\text{out},i}$ and $\mathcal{P}_{\text{srcout}|x_{\mathcal{I}}}$, we can obtain the $\mathsf{empty} \longrightarrow \mathsf{packets}$ transition probability of the output traffic. Given that the tagged output link is empty, we can obtain the probability of the switch being in system state number x via $\mathcal{P}_{x|\text{out},0}$. In order for this link to have at least one packet in the next timeslot, the input sources must transition to a next (possibly the same) input state and then they must emit at least one packet addressed to the given link. Summing over all possible system states and all possible input state transitions for each of those system states, we obtain the $\mathsf{empty} \longrightarrow \mathsf{packets}$ transition probability which is equivalent to p_{on} of the ETS source match because of the $p_0 = 0$ requirement:

$$p_{\text{on}} = \sum_{\boldsymbol{\mathcal{S}}} \left[\mathcal{P}_{x|\text{out},0} \sum_{f_{\mathcal{I}}=0}^{I} \left(\mathcal{P}_{\mathcal{I}_{\mathcal{I}}} \mathcal{P}_{\text{srcout}|f_{\mathcal{I}}} \right) \right], \tag{102}$$

where $\mathcal{P}_{\mathcal{I}_{\mathcal{I}}}$ is the probability of transitioning from input state $\mathcal{I}_{x_{\mathcal{I}}}$ ($\mathcal{I}_{x_{\mathcal{I}}} = \mathcal{S}_x^{\mathcal{I}}$) to $\mathcal{I}_{f_{\mathcal{I}}}$. By matching the $\mathsf{empty} \longrightarrow \mathsf{packets}$ probability here, we guarantee a match to β from the discussion in Section 5.3.1. Note that Equation 102 is much simpler than one describing a $\mathsf{packets} \longrightarrow \mathsf{X}$ probability because we do not have to consider the buffer state in the next timeslot (within the inner summation).

The $p_0 = 0$ requirement makes it trivial to obtain π_{off} as the source is considered to be off whenever it does not emit a packet:

$$\pi_{\text{off}} = \mathcal{P}_{\text{out},0}. \tag{103}$$

Likewise, it is easy to obtain π_{on}:

$$\pi_{\text{on}} = 1 - \pi_{\text{off}}. \tag{104}$$

Substituting Equation 103 into Equation 59 and solving for p_{off} we obtain

$$p_{\text{off}} = \frac{p_{\text{on}} \mathcal{P}_{\text{out},0}}{(1 - \mathcal{P}_{\text{out},0})}. \tag{105}$$

Finally, using $p_0 = 0$ and $\mathcal{P}_{\text{out},i}$ for q_i in Equation 65, and solving for p_i we obtain

$$p_i = \begin{cases} 0 & : \quad i = 0 \\ \frac{\mathcal{P}_{\text{out},i}}{\pi_{\text{on}}} & : \quad 0 < i \leq r \end{cases} . \tag{106}$$

Equations 102, 105 and 106 together completely describe the parameters of an ETS virtual source that matches q_i, σ, β_S and β of the switch's output link traffic.

5.3.3 Factors That Contribute to Inexactness

The interstage virtual source model matches traffic parameters that represent averages of quantitative traffic properties and cannot match some of the more complex aspects of traffic. Because the switch loss rate will typically vary in a highly nonlinear way with respect to the different "components" in the distribution of a traffic property, using an average will tend to underestimate the true loss rate because of the unequal "weight" of the components with respect to their effect on the switch (see Figure 42). The virtual source model *exactly* matches the following traffic parameters:

- the average load presented (σ);
- the temporal burstiness factor of the link traffic (β);

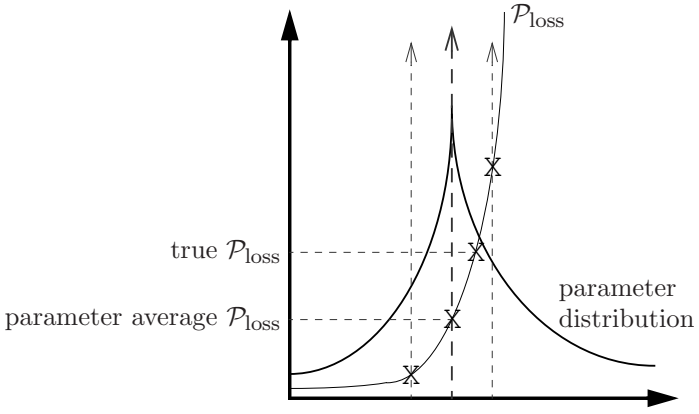


Figure 42: A nonlinear response of $\mathcal{P}_{\text{loss}}$ to variations in traffic properties contributes to errors when a parameter average is used as an approximation.

- the link occupancy probability distribution (q_i) and spatial burstiness factor (β_S);
- the average burst[5] length (λ).

Parameters that are not matched exactly include

- temporal correlations between the q_i; (For example, if a $c = 4$ link has three packets in a given timeslot, there may be a higher than normal probability that three or four packets will be in the link in the next timeslot.)
- the probability distribution of burst lengths.

Because the ETS source lacks the complexity of a switch/input source system, there are, without doubt, other traffic metrics that are not matched exactly.

5.4 Overall Loss Rate of a Network

Knowing the loss rate for the switches in each stage i ($1 \leq i \leq \eta$) of a network, it is trivial to calculate the $\mathcal{P}_{\text{loss}}$ of the overall network:

$$\mathcal{P}_{\text{loss}} = 1 - \prod_{i=1}^{\eta}(1 - \mathcal{P}_{\text{loss},i}), \tag{107}$$

where η is the number of network stages and $\mathcal{P}_{\text{loss},i}$ is the packet loss probability for switches in stage i.

However, in practice, Equation 107 performs less than satisfactorily when implemented directly in computational environments of finite floating point accuracy. With most current machines, a $\mathcal{P}_{\text{loss},i}$ less than about 10^{-15} will present problems with Equation 107 because of the need to subtract from the constant of one, which is potentially many orders of magnitude larger than $\mathcal{P}_{\text{loss},i}$. In [17, page 358], Pattavina addressed this issue and noted that, for small $\mathcal{P}_{\text{loss},i}$, the approximation

$$\mathcal{P}_{\text{loss}} \approx \sum_{i=1}^{\eta} \mathcal{P}_{\text{loss},i} \tag{108}$$

[5]Where "bursts" cannot contain empty timeslots (i.e., $p_0 = 0$) as discussed in Section 5.3.1.

is reasonably accurate. Although Equation 108 avoids the problems of Equation 107, it is an approximation and leaves it to user to decide how large an error is acceptable.

To avoid both problems, Equation 107 can be rewritten in the form

$$\mathcal{P}_{\text{loss}} = \sum_{j=1}^{\eta} (-1)^{j+1} \Upsilon \binom{\mathcal{P}_{\text{loss},i}}{j}, \tag{109}$$

where $\Upsilon\binom{\mathcal{P}_{\text{loss},i}}{j}$ is the sum of the products formed by choosing j combinations (without replacement) from the set of values $\mathcal{P}_{\text{loss},i}$ $(1 \leq i \leq \eta)$. For example, if $\eta = 3$, Equation 109 becomes

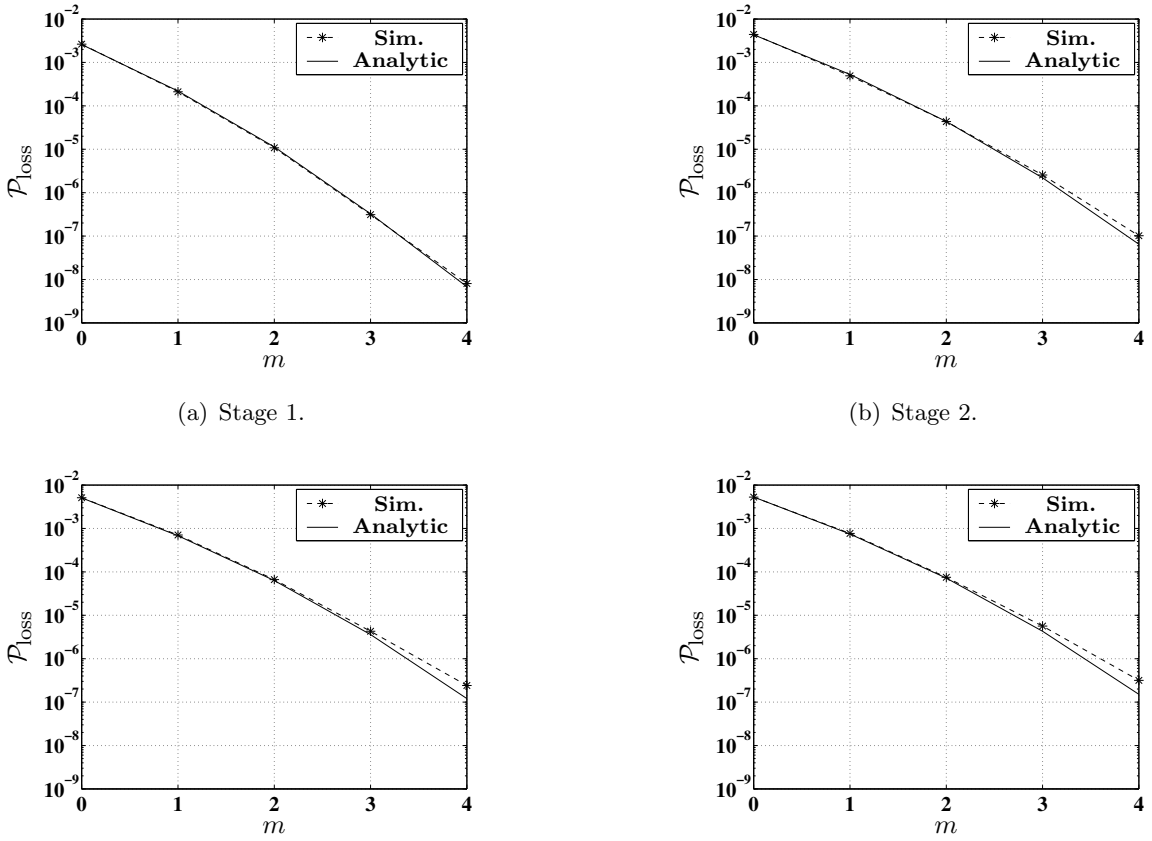
$$\begin{aligned} \mathcal{P}_{\text{loss}} = {} & \mathcal{P}_{\text{loss},1} + \mathcal{P}_{\text{loss},2} + \mathcal{P}_{\text{loss},3} \\ & - (\mathcal{P}_{\text{loss},1}\mathcal{P}_{\text{loss},2} + \mathcal{P}_{\text{loss},1}\mathcal{P}_{\text{loss},3} + \mathcal{P}_{\text{loss},2}\mathcal{P}_{\text{loss},3}) \\ & + \mathcal{P}_{\text{loss},1}\mathcal{P}_{\text{loss},2}\mathcal{P}_{\text{loss},3}. \end{aligned} \tag{110}$$

From the above, it is easy to identify Equation 108 as representing the first set of terms.

5.5 Numerical Results

Figure 43 shows the loss rates, as obtained from the analytic model and simulation, for switches at each stage of a 16×16 Banyan network composed of four stages of 2×2 switches ($c = 4$, $0 \leq m \leq 4$, $n = 2$) under a random traffic load of $\sigma = 0.3$. As mentioned in Section 3.5, throughout this work, the 95% confidence interval of the simulation results is within the asterisks unless explicit error bars indicate otherwise.

Because the model is exact for the first stage, the analytic results for stage one totally agree with those obtained from simulation. For the later stages, the analytic model slightly underestimates the loss rate. This deviance is due entirely to the use of the ETS interstage traffic approximation. Note that the loss rate of a stage 4 switch with $m = 4$ is more than an order of magnitude greater than that of a stage 1 switch with $m = 4$ because of the increase in the burstiness of the input traffic to stage 4 as a result of the use of the buffers and output channels to resolve contention in the



(a) Stage 1.

(b) Stage 2.

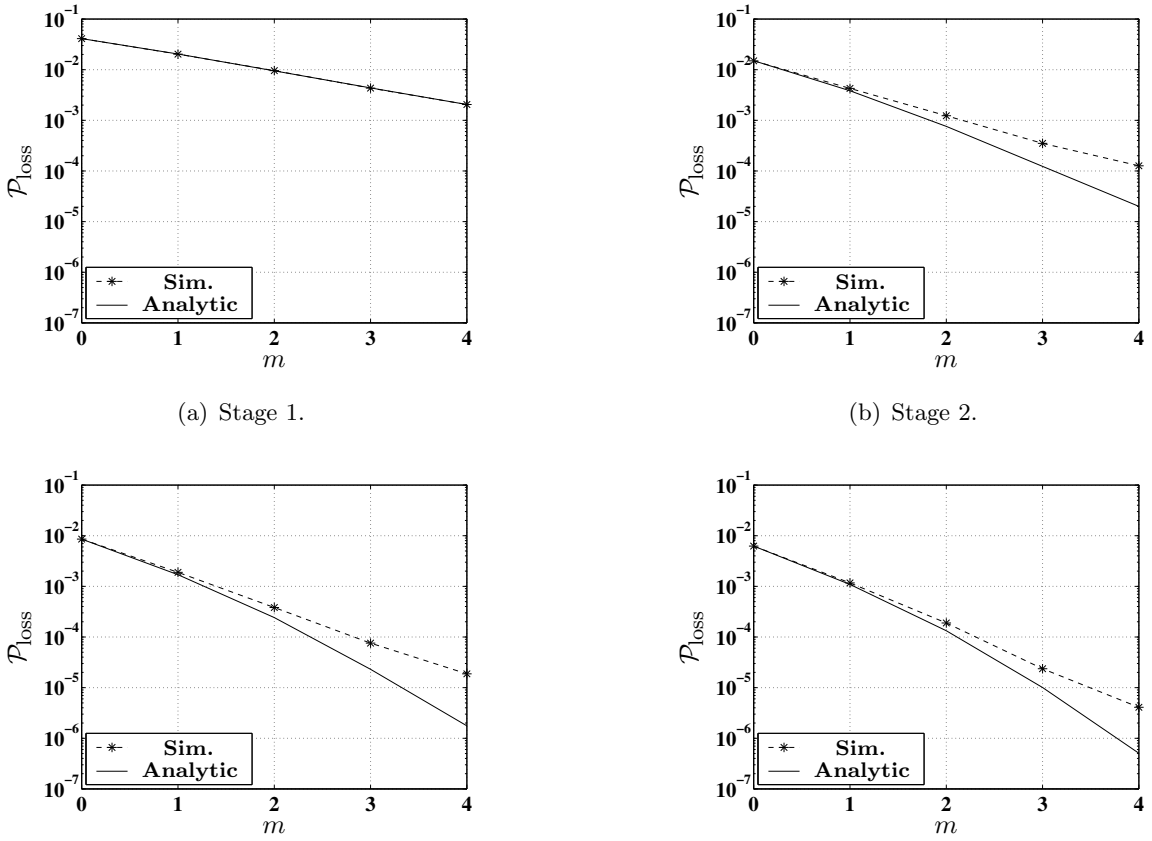
(c) Stage 3.

(d) Stage 4.

Figure 43: $\mathcal{P}_{\text{loss}}$ for switches in each stage, from both the analytic model and simulation, for a 16×16 Banyan network of 2×2 switches with $c = 4$, $0 \leq m \leq 4$ under random traffic, $\sigma = 0.3$.

earlier stages ($\beta = 1.002$, $\beta_S = 1.072$ for the input to stage 4 versus $\beta = \beta_S = 1$ for the input to stage 1).

Results for the same network under heavily bursty traffic (FOB ETS source, $\beta = 2.5$, $\beta_S \approx 2.53$, $\lambda = 4$, $p_c = 1$, $\sigma = 0.3$) are shown in Figure 44. The analytic model results deviate noticeably from those of simulation for stages after the first, especially for the lower loss rates. The increasing inaccuracy with lower $\mathcal{P}_{\text{loss}}$ values is unfortunate, but understandable — as $\mathcal{P}_{\text{loss}}$ falls, a packet loss becomes an increasingly rare event whose accurate prediction demands a more accurate modeling of the input traffic. A plausible explanation for the fact that the model tends to be less



(a) Stage 1.

(b) Stage 2.

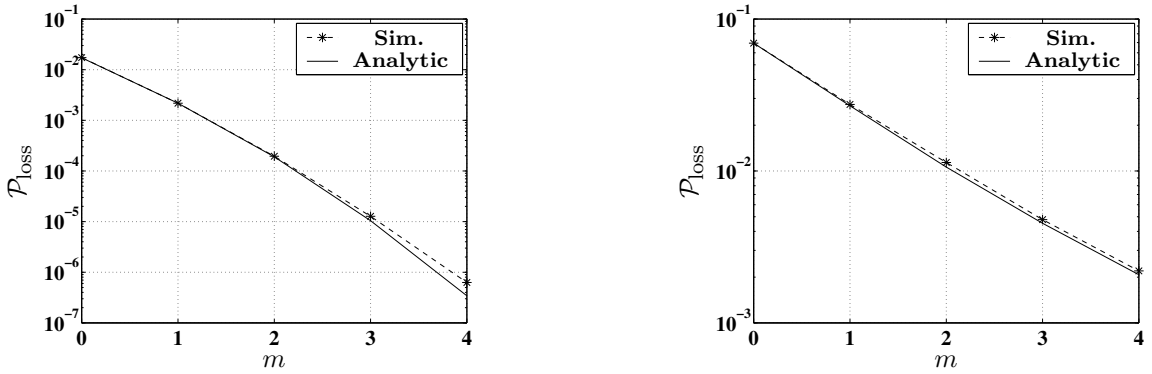
(c) Stage 3.

(d) Stage 4.

Figure 44: $\mathcal{P}_{\text{loss}}$ for switches in each stage, from both the analytic model and simulation, for a 16×16 Banyan network of 2×2 switches with $c = 4$, $0 \leq m \leq 4$ under FOB bursty traffic, $\beta = 2.5$, $\beta_S \approx 2.53$, $\lambda = 4$, $p_c = 1$, $\sigma = 0.3$.

accurate for the bursty traffic case than under the random traffic case is that the additional complexity of the two-state input sources, relative to that of single-state random input sources, makes the traffic at the output of stage 1 less conducive to accurate representation by ETS sources. For example, the inter-stage traffic may have a probability distribution of burst lengths that is quite different from the ETS virtual source, which can match only the average (as discussed in Section 5.3.3).

Perhaps surprisingly, the analytic model is extremely accurate for the overall network under both of the traffic cases as shown in Figure 45. The reason for the accurate overall loss predictions under heavily bursty traffic is because the first stage



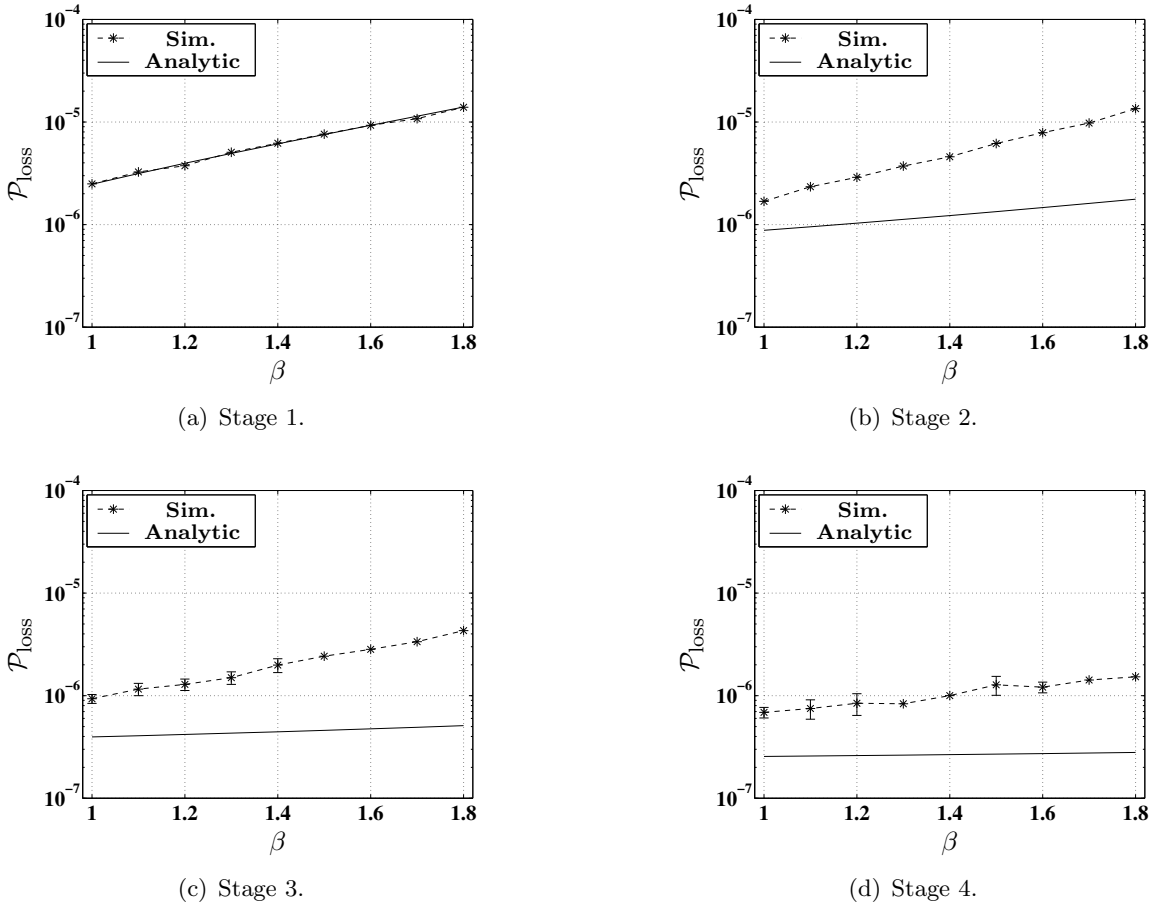
(a) Random traffic.

(b) FOB Bursty traffic.

Figure 45: $\mathcal{P}_{\text{loss}}$ of the overall 16×16 Banyan network under random and FOB bursty traffic, $\sigma = 0.3$.

is responsible for the vast majority of the loss and this stage is modeled exactly. The later stages exhibit a lower loss rate because input bursts are dispersed among all the output links of the switches (addresses of the packets in a burst are uncorrelated), which results in more random-looking traffic at the later stages.

Figure 46 shows the $\mathcal{P}_{\text{loss}}$ performance for stages in the same type of network with $c = 4$, $m = 4$ switches under traffic of moderate varying burstiness ($1 \leq \beta \leq 1.8$ yielding $2 \leq \lambda \leq 10$, $\beta_S \approx 1.56$, binomial p_i for an "on" load of $\sigma_{\text{on}} = 0.6$), and an average load of $\sigma = 0.3$. Varying β does affect the accuracy of the model somewhat, but the deviations are within the range of those of Figure 43 and Figure 44. Note that error need not accumulate as one progresses toward the later stages. The ETS virtual source used at the input to stage 2 exactly matches σ, β, β_S and q_i of the actual traffic, so inaccuracy in this stage is entirely the result of "other" traffic parameters that are not matched (Section 5.3.3). As one progresses further into the network, the interstage traffic becomes less sensitive to changes in the temporal and spatial burstiness of the input traffic to the network (Section 5.2.3), resulting in loss rates and modeling accuracy that are more constant with respect to changes in β and/or β_S (Figure 46(d)). The overall loss rate for the network is shown in Figure 47, where



(a) Stage 1.

(b) Stage 2.

(c) Stage 3.

(d) Stage 4.

Figure 46: $\mathcal{P}_{\text{loss}}$ for switches in each stage, from both the analytic model and simulation, for a 16×16 Banyan network of 2×2 switches with $c = 4$, $m = 4$ under moderately bursty traffic, $1 \leq \beta \leq 1.8$, $2 \leq \lambda \leq 10$, $\beta_S \approx 1.56$, binomial p_i (for $\sigma_{\text{on}} = 0.6$), $\sigma = 0.3$.

model accuracy is also reasonably well behaved with respect to varying β.

Finally, the model also performs well for larger (64×64) networks composed of larger switches ($c = 1$, $m = 10$, $n = 8$) under mildly bursty ($\beta = 1.25$) traffic of varying λ ($3 \leq \lambda \leq 12$) and average load of $\sigma = 0.5$ as shown in Figures 48 and 49. Figure 48(b) indicates that variations in λ (with β held constant) do not appear to adversely affect model accuracy despite the $p_0 = 0$ ETS match requirement (Section 5.3.1).

The analytic model can be used to analyze networks that are impractical to

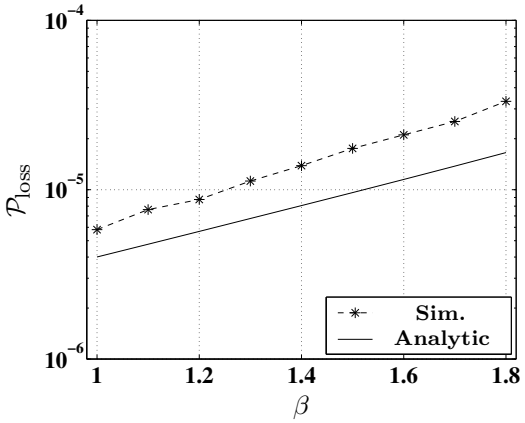
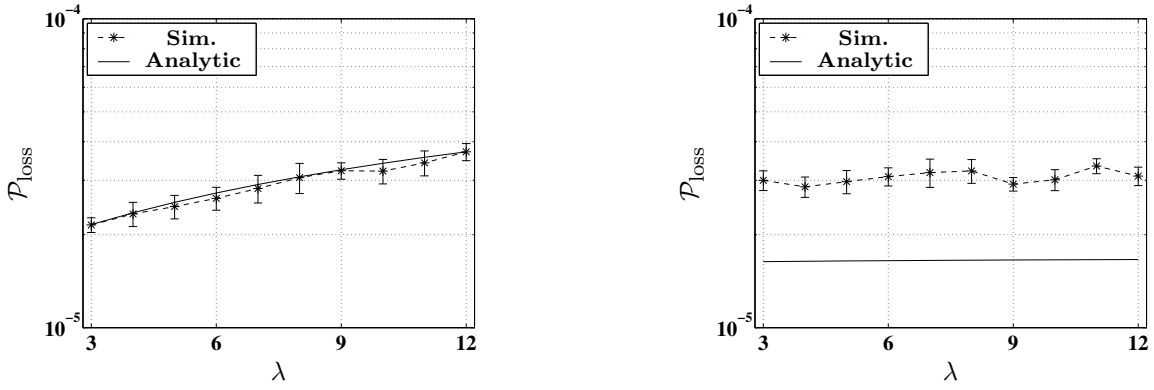


Figure 47: $\mathcal{P}_{\text{loss}}$ for the overall 16×16 Banyan network under moderately bursty traffic, $1 \leq \beta \leq 1.8$, $\beta_S \approx 1.56$, $\sigma = 0.3$.



(a) Stage 1.

(b) Stage 2.

Figure 48: $\mathcal{P}_{\text{loss}}$ for switches in each stage, from both the analytic model and simulation, for a 64×64 Banyan network of 8×8 switches with $c = 1$, $m = 10$ under bursty traffic of varying burst lengths, $\beta = 1.25$, $3 \leq \lambda \leq 12$, $\sigma = 0.5$.

simulate, either because the loss rate is too low and/or the network is too large. The latter situation can result because of the fact that the computational effort required by the analytic model (for a given switch size) grows linearly with an increasing number of network stages (η) while the size of the network to be simulated grows exponentially (n^{η}). Such very large or low loss networks were not considered here because comparison with results from simulation would not be viable — not because

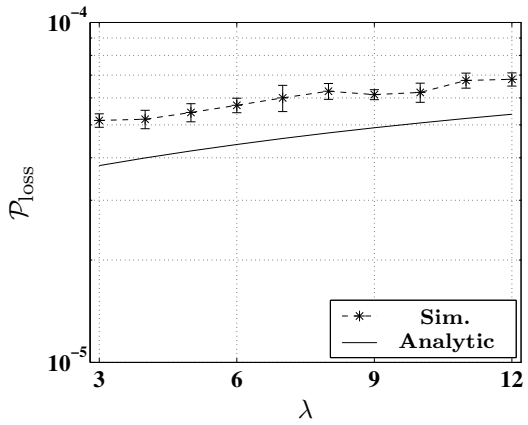


Figure 49: $\mathcal{P}_{\text{loss}}$ for the overall 64×64 Banyan network under bursty traffic of varying burst lengths, $\beta = 1.25$, $3 \leq \lambda \leq 12$, $\sigma = 0.5$.

of limitations of the analytic model.

It appears that the accuracy of the presented model is generally superior to other shared-memory packet switch network models in published literature [24], [56], [58], [63]. Direct comparisons are not possible because the published models are for ESMP switches, use different traffic models or network parameters, are not general enough to be applicable to switches with channel-grouped links and did not provide results for the individual network stages.

Chapter 6

Asymmetric Switches

It is sometimes useful or necessary to employ switches that are asymmetric with respect to the number of input and output links. Switches that have more input than output links ($h > g$) are known as "concentrators" or "multiplexers." Conversely, switches with more output than input links ($g > h$) are known as "expanders" or "demultiplexers." Such asymmetry has a large effect on switch performance and can alter the properties of the traffic in the output links, including load, in ways not possible with symmetric switches.

Figure 50 (solid curve) shows the $\mathcal{P}_{\text{loss}}$ rates for $c = 1$, $g = 4$, $m = 12$ switches under a random traffic load of $\sigma = 0.5$ as a function of h. Increasing h from 2 (expander) to 8 (concentrator) results in a 20 order-of-magnitude increase in $\mathcal{P}_{\text{loss}}$. Such a large range is caused by the fact that increasing h results in an increase in the total bandwidth carried by the switch as well as the total number of packets that can simultaneously arrive at the switch. Once again, this raises the issue of fairness because the $h = 8$ switch is being asked to handle four times the bandwidth of the

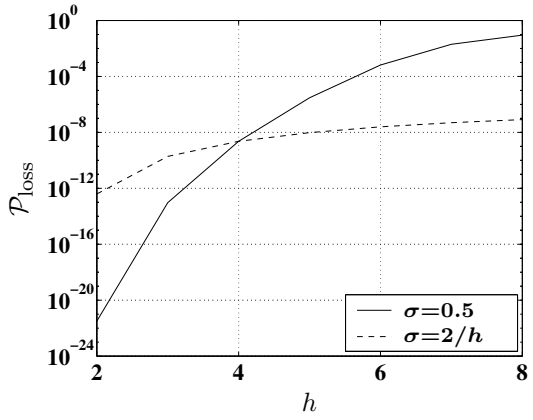


Figure 50: $\mathcal{P}_{\text{loss}}$ for $c = 1$, $g = 4$, $m = 12$ switches under random traffic of $\sigma = 0.5$ and $\sigma = \frac{0.5g}{h} = \frac{2}{h}$ as a function of h ($2 \leq h \leq 8$).

$h = 2$ switch. If instead, the input link load is adjusted so that the total switched bandwidth is held constant with respect to h ($\sigma = \frac{0.5g}{h} = \frac{2}{h}$), $\mathcal{P}_{\text{loss}}$ increases less dramatically with increases in h (Figure 50, dashed curve). The increase in $\mathcal{P}_{\text{loss}}$ with h is because of the fact that having more input links allows for more packets to arrive simultaneously to the switch — events that can be severely detrimental to $\mathcal{P}_{\text{loss}}$ performance even if they happen relatively infrequently.

Varying g also greatly affects switch performance, but with some differences from h/g asymmetry induced by varying h. Figure 51 (solid curve) shows the $\mathcal{P}_{\text{loss}}$ rates for $c = 1$, $h = 4$, $m = 12$ switches under a random traffic load of $\sigma = 0.5$ as a function of g. The nearly 15 order-of-magnitude reduction in $\mathcal{P}_{\text{loss}}$ as g is increased from 2 to 8 is somewhat less of a range than when h is varied by a comparable amount. Increasing g does reduce the probability of output port contention, but unlike variations with respect to h, it does not affect the total bandwidth input to the switch or the maximum number of packets that can arrive simultaneously.

However, the total output capacity of the switch is determined by g. If instead, the input link load is adjusted so that the total input bandwidth remains matched to the total output link capacity so that the output link load (neglecting the effect

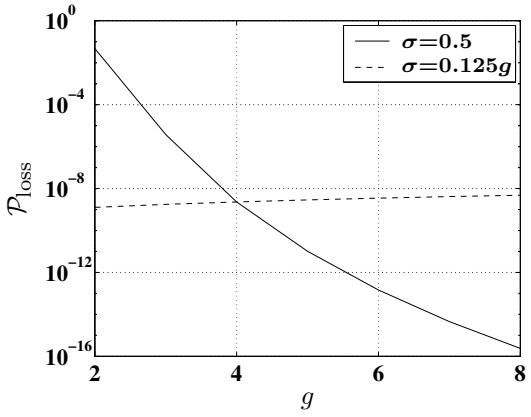


Figure 51: $\mathcal{P}_{\text{loss}}$ for $c = 1$, $h = 4$, $m = 12$ switches under random traffic of $\sigma = 0.5$ and $\sigma = \frac{0.5}{h/g} = 0.125g$ as a function of g ($2 \leq g \leq 8$).

of packet loss on output link load) is held constant at 50% ($\sigma = \frac{0.5}{h/g} = 0.125g$), $\mathcal{P}_{\text{loss}}$ becomes practically insensitive to h/g asymmetry as shown by the dashed curve in Figure 51. Under such circumstances, this atypical increase in $\mathcal{P}_{\text{loss}}$ with g is likely because of the fact that the total switched bandwidth increases linearly with g while the buffer size is held constant. Barring such unusual adjustments to the input load, it is generally true that concentrators have higher loss rates than their expander counterparts.

Of course, severe packet losses are unavoidable if the output links are physically incapable of handling the total input bandwidth — there is often a heavy price to be paid for trying to do, or even just approach, the impossible. For instance, the $c = 1$, $h = 4$, $g = 2$, $m = 12$, $\sigma = 0.5$ switch in Figure 51 (solid curve) has a $\mathcal{P}_{\text{loss}}$ of 4.7×10^{-2}. This improves very little even with drastic increases in buffer size ($\mathcal{P}_{\text{loss}} = 1.4 \times 10^{-2}$ for $m = 48$) because of the futility of trying to get the output links to carry a load approaching 100%.[1] If g is reduced further to $g = 1$, at least half of the input packets would have to be lost regardless of the buffer size.

Finally, asymmetric channel grouping factors (Section 3.5) can be used in conjunction with, and possibly to compensate for, h/g asymmetry. For example, an $h = 4$, $g = 2$, $s = 1$, $r = 2$, $m = 12$ switch under a random traffic load of $\sigma = 0.5$ has a very respectable $\mathcal{P}_{\text{loss}} = 2.1 \times 10^{-13}$ which is an 11 order-of-magnitude reduction from that of the corresponding $r = 1$ case of Figure 51 ($\mathcal{P}_{\text{loss}} = 4.7 \times 10^{-2}$).

6.1 Trill Networks

In order to properly describe networks that utilize switches of differing parameters (in different stages) it is necessary to expand the parameter terminology. Let c_i, g_i, h_i, m_i, n_i, r_i, s_i represent their respective (nonindexed) switch parameters for switches in stage i ($1 \leq i \leq \eta$) of a network.

[1] It is theoretically possible to have an output load of 100%, but only in the absence of output link contention.

Liew and Lu [43], [44], [53] have proposed a general three stage network architecture that allows the construction of very large packet networks (asymmetric or symmetric) out of much simpler switches. Liew and Lu did not name their design. "Trill" seems appropriate for "Tri-stage Liew and Lu" and will be used to refer to such networks in this work. The Trill architecture is shown in Figure 52. Switches in the first two stages are organized into logical groups called "partitions" (indicated as dashed boxes in Figure 52). Because the h_3 stage 1-2 partitions are not interconnected, they may have a physical basis with each corresponding to a self-contained physical switch "module." The ith stage 3 partition contains switches that handle the output traffic from the ith second stage switch in each stage 1-2 partition. Because the switches within a stage 3 partition are not interconnected, a physical basis of a stage 3 partition is less clear than that of the stage 1-2 partitions.

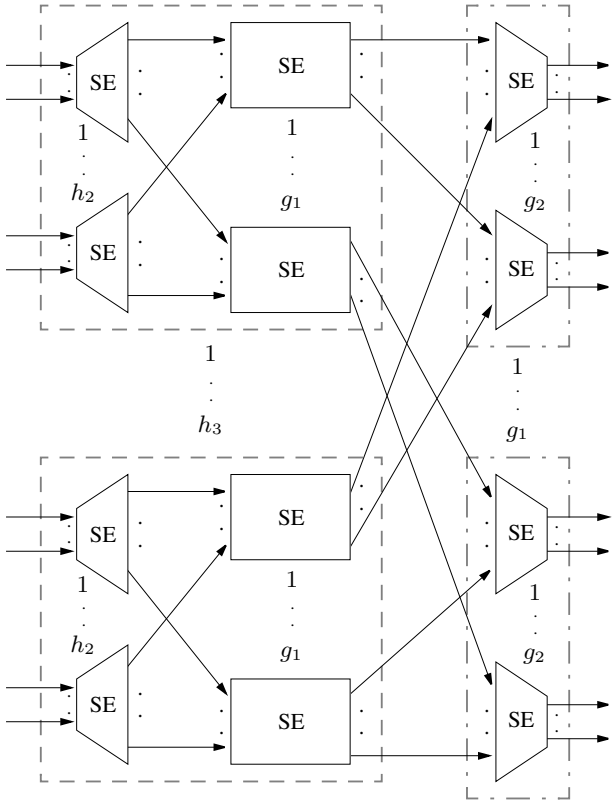


Figure 52: The Trill architecture.

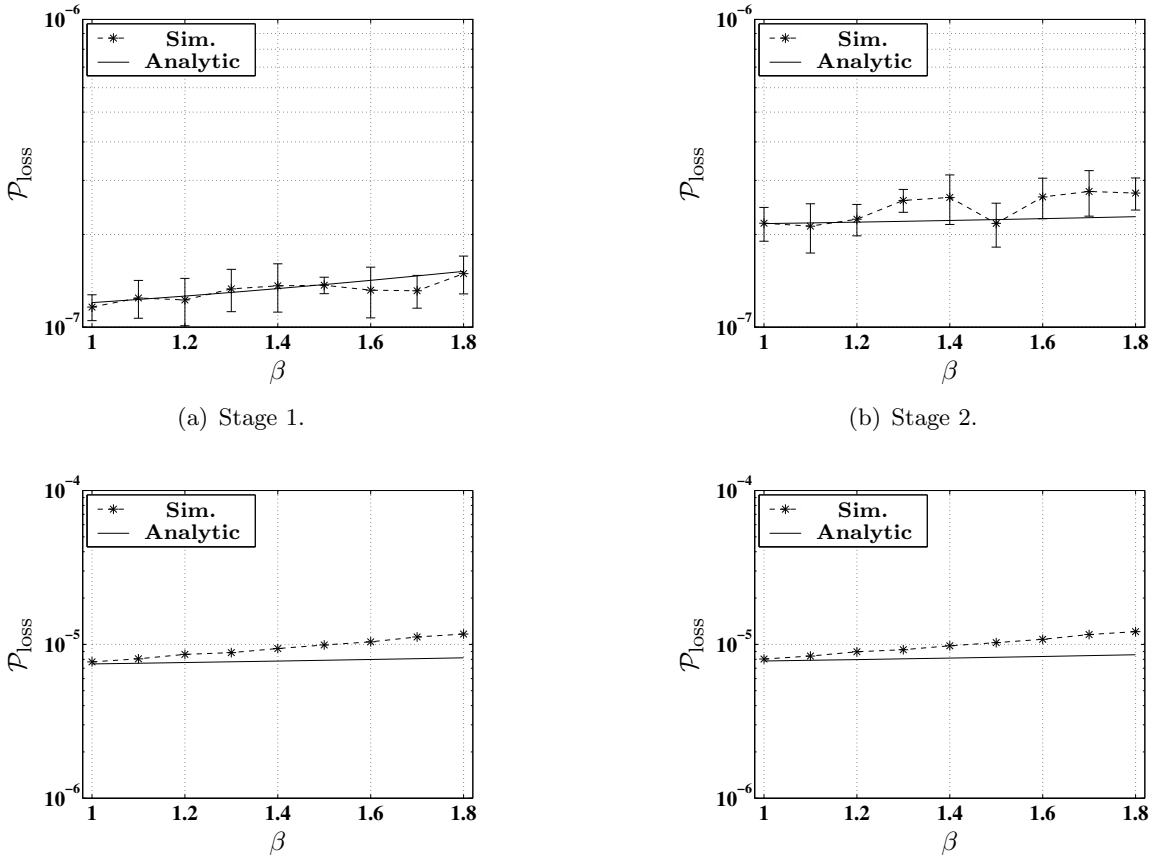
The switches in stages 1, 2 and 3 need not necessarily be expanders, symmetric, and concentrators, respectively, though this is the typical configuration suggested and analyzed by Liew and Lu [53]. The first stage switches select which of the g_1 stage 3 partitions a given packet is destined to. The second stage switches select from the g_2 switches within the given stage 3 partition. Each stage 3 switch then routes packets to one of its g_3 output links.

Trill networks satisfy the requirements for analysis by the virtual source approach of the analytic model of this work (Section 5.1). In fact, Trill networks are a generalization of a member of the three stage Banyan family. This can easily be seen by observing that if 2×2 switches are used for all three stages, Figure 52 reduces to Figure 8, although with the stage 1-2 and stage 2-3 interconnections transposed from input to output.

Some basic properties of the overall Trill network can be written in terms of the parameters of its switches (refer to Figure 52). The number of input links is given by $h_1 h_2 h_3$. The number of output links is given by $g_1 g_2 g_3$. The number of 1-2 and 2-3 interstage links is given by $g_1 h_2 h_3$ and $g_1 g_2 h_3$, respectively. The number of switches in stage 1, 2 and 3 is given by $h_2 h_3$, $g_1 h_3$ and $g_1 g_2$, respectively. Finally, Liew and Lu defined [53] the "expansion ratio" as the number of 1-2 interstage links divided by the number of input links ($\frac{g_1}{h_1}$).

6.1.1 Numerical Results

Trill networks can sometimes outperform their Banyan equivalents. Figure 53 shows $\mathcal{P}_{\text{loss}}$ for each stage as well as the overall network for a 16×16 Trill network of 2×4, 2×2 and 4×2 switches (in stages 1, 2, and 3, respectively) with $c = 4$, $m_1 = m_2 = 3$ and $m_3 = 4$ under moderately bursty traffic, $1 \leq \beta \leq 1.8$ yielding $2 \leq \lambda \leq 10$, $\beta_S \approx 1.56$, binomial p_i for an "on" load of $\sigma_{\text{on}} = 0.6$, and an average load of $\sigma = 0.3$. This is the same input traffic used for the four-stage 16×16 Banyan of 2×2 switches of Figure 46 and Figure 47. The Trill network uses 32 switches (eight 2×4, sixteen



(a) Stage 1.

(b) Stage 2.

(c) Stage 3.

(d) Overall Network.

Figure 53: $\mathcal{P}_{\text{loss}}$ for switches in each stage and the overall network, from both the analytic model and simulation, for a 16×16 Trill network of 2×4, 2×2 and 4×2 switches with $c = 4$, $m_1 = m_2 = 3$, and $m_3 = 4$ under moderately bursty traffic, $1 \leq \beta \leq 1.8$, $2 \leq \lambda \leq 10$, $\beta_S \approx 1.56$, binomial p_i (for $\sigma_{\text{on}} = 0.6$), $\sigma = 0.3$.

2×2 and eight 4×2) versus the 32 switches (all 2×2) required by the Banyan. Although the Trill network does use larger 2×4 and 4×2 switches, the total number of buffer cells in the network, 104, is less than the 128 employed by the Banyan. However, the Trill network exhibits overall loss rates similar to that of its Banyan counterpart with the former even outperforming the latter for $\beta \geq 1.3$ as can be seen by comparing Figure 53(d) with Figure 47. The Trill network accomplishes this by using 2×4 expanders in the first stage and twice the number of 2×2 switches in the second stage than that of the Banyan. Thus, the Trill network can obtain lower

loss rates in the first stage via the use of expanders and in the second stage via the use of a greater than normal number of switches with each of these handling half the bandwidth of its Banyan counterpart — thus allowing for a reduction in the number of buffer cells used by switches in these stages. This strategy can pay off when the highest loss rates are found in the first stages, which is typically the case under bursty traffic.

However, expansion ratios greater than unity do not always yield better loss rates, especially if the total number of buffer cells used by the network is fixed. Table 3 shows the performance, as determined by the analytic model for $c = 1$, 128×128 Trill networks with expansion ratios of 1, 2 and 4, under a random traffic load of $\sigma = 0.30$. The total number of buffer cells used in each stage is held constant with each network having a total of 832 buffer cells.[2] As can be seen in Table 3, increasing the expansion ratio not only dramatically increases the total number of switches and interstage links required, it results in much worse loss rates. It may be noted that, in the networks with non-unity expansion ratios, the number of buffer cells used in each stage has not been optimally allocated. Specifically, in the network with an expansion ratio of two, it is advisable to decrease the number of buffer cells in stage 1 switches and apply these to the later stages where they are more badly needed. However, there are simply not enough "excess" buffer cells in stage 1 to be able to reduce the later stages' loss rates to the 10^{-10} range needed to match the performance of the unity expansion ratio network. For example, if two buffer cells are taken from each of the 32 stage 1 switches and applied to the 64 stage 3 switches so that each stage 3 switch has an additional buffer cell ($m_3 = 7$), $\mathcal{P}_{\text{loss},1}$ becomes 2.2×10^{-11}, $\mathcal{P}_{\text{loss},2}$ becomes 1.7×10^{-9} and $\mathcal{P}_{\text{loss},3}$ becomes 1.8×10^{-8} yielding an overall $\mathcal{P}_{\text{loss}}$ of 1.9×10^{-8}, which is not low enough to match the performance of the network with unity expansion ratio. The situation becomes even worse as the expansion ratio is increased to four.

[2]That is, the number of buffer cells in switches in each stage is decreased proportionally with increases to the number of switches in that stage.

Table 3: Performance, as determined by the analytic model, of $c = 1$, 128×128 Trill networks with expansion ratios of 1, 2 and 4, under a random traffic load of $\sigma = 0.30$. In each of the three cases, there is a total of 832 buffer cells in the network.

	Expansion Ratio		
	1	2	4
Number of Interstage Links	128	256	512
SE Size, Stage #1 ($h_1 \times g_1$, m_1)	4×4, $m_1 = 8$	4×8, $m_1 = 8$	4×16, $m_1 = 8$
SE Size, Stage #2 ($h_2 \times g_2$, m_2)	8×8, $m_2 = 12$	8×8, $m_2 = 6$	8×8, $m_2 = 3$
SE Size, Stage #3 ($h_3 \times g_3$, m_3)	4×4, $m_3 = 12$	4×2, $m_3 = 6$	4×1, $m_3 = 3$
Number of SE, Stage #1	32	32	32
Number of SE, Stage #2	16	32	64
Number of SE, Stage #3	32	64	128
Total Number of SE	80	128	224
$\mathcal{P}_{\text{loss}}$, Stage #1	2.7×10^{-10}	1.2×10^{-14}	1.2×10^{-18}
$\mathcal{P}_{\text{loss}}$, Stage #2	1.3×10^{-11}	1.7×10^{-9}	3.2×10^{-7}
$\mathcal{P}_{\text{loss}}$, Stage #3	1.8×10^{-13}	1.7×10^{-7}	1.2×10^{-4}
$\mathcal{P}_{\text{loss}}$, Overall	2.8×10^{-10}	1.7×10^{-7}	1.2×10^{-4}

All things being equal, a larger number of switches is undesirable, not only because of the added complexity, but because the shared-buffer concept is partially undermined by the use of multiple switches because different switches cannot share buffer cells globally among themselves as needed. Thus, it is generally desirable, from a loss rate perspective, to use larger and fewer switches when constructing networks if the total number of buffer cells is to be minimized for a given maximum acceptable loss rate.

Chapter 7

Conclusion

Analytical modeling can be useful not only to predict the performance of systems for which simulation is not practical, but also to help in identifying and understanding underlying mechanisms and design trade-offs. There are several general principles and insights that can be observed from the results of this work.

Although exact modeling may not be suitable for larger switches, maintaining an exact model of the internal operation of a switch can pay off in terms of network modeling accuracy. This is true in spite of the fact that the ETS approximation of the interstage traffic is simplistic. Quantitative traffic properties σ, β, β_S and λ were defined and shown to be predictive of switch performance under ETS traffic. λ in itself generally has less of an effect on packet loss relative to the other properties.

Small buffer sizes pose serious difficulties in obtaining low packet loss rates. Furthermore, additional buffer cells tend to have more of an effect at lower loads. So, all-optical networks with severe buffer limitations will likely have to be operated at modest loads or used with transport protocols that are tolerant of higher packet loss rates than are typically encountered in current systems.

Channel grouping has the potential to reduce the need for buffering, but it should be evaluated against alternatives such as speeding up the links. Using multichannel links to resolve contention can degrade performance in downstream switches because of spatial burstiness. This can happen even in bufferless systems. Likewise, buffering can result in increases in the temporal burstiness of a switch's output traffic. What is of benefit to an isolated switch may have detrimental effects within the context of a network.

Care must be taken when evaluating the effects of channel grouping and/or asymmetry because their net benefits often depend on how a fair comparison is defined.

In some situations, network designs that use asymmetric switches are superior to their symmetric-only counterparts. Finally, generally speaking, if the total number of buffer cells is to be minimized, it is better to use larger, but fewer, switches when designing a network so as to reap the maximum benefit from the shared memory scheme.

7.1 Contributions of Work

As mentioned in Section 1.1.1, this work offers several original contributions. These will now be briefly reviewed.

More capable input traffic models: The use of the ETS traffic model provides a means to analyze OSMA switches under (temporally) bursty traffic or traffic with arbitrary occupancy probability distributions (spatial burstiness). Previous OSMA models were confined to analysis of random traffic with a binomial occupancy distribution.

Network analysis: The model of this work can be used to analyze networks of OSMA switches through the use of virtual sources. Previous OSMA models were applicable only to single switches.

Asymmetric switches: The developed model is general enough to be used with asymmetric (with respect to number of ports and/or channel grouping factors of the inputs and outputs) OSMA switches. Previous OSMA models were applicable only to symmetric switches.

Computationally efficient: The buffer state and arrival vector reduction techniques in this work are more advanced than those of previous OSMA models. Thus, switches can be analyzed with less computational effort.

7.2 Suggestions for Future Research

One of the enduring aspects of engineering is that there are always more problems to solve, improvements to be made and hidden truths to be found. Here are some of the

more prominent "unturned stones" of this work that have the potential to become fruitful areas of investigation.

Cost-benefit and/or optimization analysis: The results of this work open the door to a large number of optimization issues that were deliberately avoided because they involve specific judgments to be made concerning the actual costs associated with implementing the different alternatives. How many buffer cells is a channel worth? What is the cost of a lost packet in terms of buffer cells? Up to what point should load be sacrificed for lower loss rates? Once questions such as these are addressed, it becomes possible to investigate an optimal solution. There are also optimization issues related to the network as a whole, such as how to best allocate channel-grouping factors, buffer cells, etc.

Traffic model improvements: Network modeling accuracy could be improved via the use of a more complex model of interstage traffic. Traffic closer to end users is likely to have strong correlations and/or non-uniformities in the packet addresses. New buffer state and arrival vector reduction techniques would have to be developed in order to handle such traffic.

More aggressive state space reductions: Improved state space and/or arrival vector reduction techniques would allow the analysis of larger switches. If exactness is not required, novel approximate models could be pursued, such as retaining only the first i terms of the buffer state of the exact model.

Properties and ramifications of "asymptotic" network traffic: In Section 5.2.3 the concept of "eigentraffic" in networks was introduced. It might be interesting to predict the properties of traffic deep within a network based solely on the analysis of a single switch. Large networks could then be analyzed and optimized without the need of multiswitch analysis or simulation.

Use of traffic conditioning devices: External "in-line" devices could be used to condition link traffic so as to reduce contention within the switches. Bufferless "feed-forward" switches such as Haas' Staggering switch might be useful in this regard.

Revised switch designs/operation: A hybrid OSMA design could be proposed that incorporates ideas from Karol's SMOP and Haas' Staggering switches. Specifically, longer, unequal delay lines could be used to hold packets that are known to contend in the *next* timeslot. However, OSMA style (single timeslot delay) buffer cells would be used during most "typical" operation for low latency and better packet loss performance on a per-buffer-cell basis. Also, the OSMA switch routing protocols could be modified in ways that improve the properties of the output traffic from the point of view of downstream switches.

Alternative technologies to reduce or to better tolerate packet loss: Instead of simply dropping packets, OSMA switches could be configured to route them to a different output link within a network topology that may allow the misrouted packets to eventually find their destination, a concept known as deflection (aka "hot-potato") routing [71]. Finally, block erasure codes could be used to introduce redundancy at the packet level which would allow end users to mitigate the loss of information contained in one or more packets [72], [73].

Appendix A

Partitions of Integers

As discussed in Section 3.2.3, the buffer states of the model of this work are represented by sets of partitions of integers. A partition of an integer is a division of the integer into positive parts without regard to ordering of the parts.[1] Partitions are typically written with the parts in decreasing order. For example, there are five unrestricted partitions of the integer four: 4, $3+1$, $2+2$, $2+1+1$ and $1+1+1+1$.

The theory of partitions of integers has an interesting history. In the mid-eighteenth century, Leonhard Euler, one of the important founders of modern mathematics, was the first to show that the number of unrestricted partitions of an integer n, which is traditionally denoted[2] in mathematics as $p(n)$, could be found by solving for the coefficients of polynomials obtained from generating functions [76]:

$$\sum_{n=0}^{\infty} p(n)x^n = \prod_{i=1}^{\infty} \frac{1}{1-x^i}. \tag{111}$$

$p(n)$ can also be determined from recursive relations [75].

In 1918, the legendary mathematician Srinivasa Ramanujan, along with Cambridge mathematician Godfrey Hardy who brought Ramanujan to the attention of the Western world, developed an asymptotic closed-form expression for $p(n)$ that approaches the true value for large n [77]. (The Hardy-Ramanujan formula was also independently discovered by James Uspensky in 1920 [78].) In 1937, Hans Rademacher improved on Ramanujan and Hardy's work by developing an exact solution for $p(n)$ [79], [80]. Andrew Odlyzko has shown that the Hardy-Ramanujan-Rademacher equation

[1] For a more formal and detailed treatment of the topic of partitions of integers, the reader is referred to [74] and [75].

[2] In this appendix, some concessions have to be made with regard to notation in order to accommodate fields with other conventions. In particular, $p(n)$ and n are used here in their conventional mathematical sense and not as they are (or might have been) defined in the equations of the previous chapters. It should be obvious from context which notation is in effect.

is a nearly optimal way to compute $p(n)$ for large n [75]. However, for the purposes of this work (small n), the Hardy-Ramanujan-Rademacher equation is quite complex and unwieldy to employ. Furthermore, the model of this work requires restrictions on the length and element (part) values of the partitions as well as requires the *names* of the partitions. So, unfortunately $p(n)$ is of little use here. A more direct computational approach has to be employed.

To address these requirements, the operator $\mathbf{\Psi}$ was introduced in Section 3.2.3. Its definition and basic properties will now be repeated here for convenience. $\mathbf{\Psi}(k)$ generates the set of unrestricted partitions of the nonnegative integer k. $\mathbf{\Psi}(k, \omega)$ is the set of partitions of k not exceeding ω in length. $\mathbf{\Psi}(k, \omega, \theta)$ has the further restriction that each element not exceed θ. Thus, $\mathbf{\Psi}(k) = \mathbf{\Psi}(k, k) = \mathbf{\Psi}(k, k, k)$. A partition of k may have length less than ω. In such cases, $\mathbf{\Psi}(k, \omega)$ is defined to append zeros as needed, to ensure that the returned partitions always have ω elements. This is done purely for computational convenience. $\mathbf{\Psi}_i(k)$ designates the ith partition and $\Psi_{i,j}(k)$ is the jth element in the ith partition ($0 \leq i < |\mathbf{\Psi}(k)|$, $0 \leq j < \omega = k$). $|\mathbf{\Psi}(k)|$ is analogous to $p(n)$ for $n = k > 0$. The parameters k, ω and θ are always nonnegative. The boundary conditions for these parameters at zero are listed below in order of decreasing precedence:

$$|\mathbf{\Psi}(k, 0, \theta)| = 0, \qquad \text{for } k \geq 0,\ \theta \geq 0, \tag{112}$$

$$|\mathbf{\Psi}(0, \omega, \theta)| = 1, \qquad \text{for } \omega > 0,\ \theta \geq 0, \tag{113}$$

$$|\mathbf{\Psi}(k, \omega, 0)| = 0, \qquad \text{for } k > 0,\ \omega > 0. \tag{114}$$

It should be pointed out that Equation 112 and Equation 113 deviate from what is usually done by convention in mathematics. Specifically, Equation 112 forbids the partitioning of zero into zero parts and Equation 113 allows the partitioning of zero into one or more parts. The latter is needed to describe the single state of the degenerate case of a bufferless switch. However, Andrews [74, page 1] and Knuth [75] state

that there is one partition of zero into exactly zero parts and no partition of zero into exactly one or more parts.

These differing conventions can be reconciled somewhat by taking the viewpoint that $\mathbf{\Psi}(0,0)$ exists, but that its name cannot be spoken! That is to say, to handle $\mathbf{\Psi}(0,\omega)$ with $\omega > 0$, the ω zero "placeholders" can be used as a surrogate state name, but being denied even this option with $\mathbf{\Psi}(0,0)$, the partition cannot be rendered and thus cannot be counted with $|\mathbf{\Psi}(0,0)|$.[3] This issue is of importance only in the degenerate boundary cases, but is mentioned here in the event the $\mathbf{\Psi}$ operator, as defined here, is used for, or compared to, related operators from other work.

The following two functions, written in C, will implement an algorithm that provides the full functionality of $\mathbf{\Psi}(k,\omega,\theta)$:

```
/* firstpart() and nextpart() integer partitioning functions
  (c) Michael Shell 1999-2004
  Released under the BSD license (2004 version)
  http://www.opensource.org/licenses/bsd-license.php
  and may be freely used, distributed and modified.
  This code is offered as-is without any warranty either expressed or
  implied, without even the implied warranty of MERCHANTABILITY or
  FITNESS FOR A PARTICULAR PURPOSE.
  usage:
  firstpart(integer_to_partition, num_partition_elements,
            max_element_value, pointer_to_partition_array, offset)
  nextpart(max_partition_length, pointer_to_partition_array, offset)
  return values:
  0 = operation not successful, no valid partition found
```

[3] Concepts such as this can have profound philosophical implications — the world of mathematical logic and description is not confined within the narrower boundaries of the world of finite computation and representation.

```
  1 = partition array has a valid partition */
int firstpart(unsigned int inttopart, unsigned int numel,
              unsigned int maxvalue, unsigned int * partarray,
              unsigned int offset)
{
unsigned int i;         /* counter */
unsigned int remaining; /* amount of integer remaining to partition */
/* zero partition array */
/* could use Unix memset() instead */
for (i=offset;i<(numel+offset);i++) partarray[i] = 0;
/* must have at least one element */
if (numel == 0) return(0);
/* allow a partition of zero into more than zero parts */
if (inttopart == 0) return(1);
/* try to create partition */
remaining = inttopart;
for (i=offset;i<(numel+offset);i++)
    {
    if (remaining > maxvalue)
       {
       partarray[i] = maxvalue;
       remaining -= maxvalue;
       continue;
       }
    else
       {
       partarray[i] = remaining;
       remaining = 0;
```

```
       break;
       }
    }
/* successful if remaining is zero */
if (remaining == 0) return(1);
else return(0);
}

int nextpart(unsigned int numel, unsigned int * partarray,
             unsigned int offset)
{
unsigned int i,ip,j,k;    /* counters */
unsigned int sum;         /* sum of right hand side (RHS) elements */
unsigned int prev;        /* previous element value */
unsigned int maxvalue;    /* maximum value an element can have */
unsigned int totaltodist; /* total to be redistributed */
/* must have more than one element to have a second partition */
/* catch potential out of range array access if numel == 0 */
if (numel < 2) return(0);
/* initialize */
prev = 0;
sum = 0;
/* start scanning from the RHS */
/* use ip = i+1 as unsigned int cannot be negative */
for (ip=(numel+offset);ip>offset;ip--)
    {
    i = ip - 1;
    /* continue scanning until find a possible redistribution point */
```

```
if (partarray[i] == prev || partarray[i] == 1)
   {
   sum += partarray[i]; /* sum of elements scanned */
   prev = partarray[i]; /* element now is the previous element */
   continue;
   }
/* partarray[i] > 1 and partarray[i] != prev */
maxvalue = partarray[i] - 1; /* value elements cannot exceed */
totaltodist = sum + 1; /* compensate for decrement of element */
/* if redistribution is not possible here,
   back out and continue scanning */
if (totaltodist > ((numel+offset)-(i+1))*maxvalue)
   {
   sum += partarray[i];
   prev = partarray[i];
   continue;
   }
/* redistribution is possible */
/* decrement this element by one and redistribute totaltodist */
partarray[i] = partarray[i] - 1;
/* redistribute and create sub-partition */
for (j=i+1;j<(numel+offset);j++)
    {
    if (totaltodist > maxvalue)
       {
       /* single element cannot hold all */
       partarray[j] = maxvalue;
       totaltodist -= maxvalue;
```

```
            }
         else
            {
            /* single element can hold all */
            partarray[j] = totaltodist;
            /* zero remaining elements */
            for (k=j+1;k<(numel+offset);k++)
                {
                partarray[k] = 0;
                }
            /* successful redistribution */
            return(1);
            }
         } /* end redistribution */
    } /* end scanning */
/* unable to find a valid redistribution, no valid next partition */
return(0);
}
```

These two functions utilize an offset parameter to allow for the independent manipulation of multiple partitions that are contained within the single array `part[]`. This capability is useful when working with multiple groups of partitions at once, such as is done with the arrival groups of Section 3.3.3.

A skeleton example of use is shown here:

```
unsigned int gotpartition;
/* obtain parameters and allocate needed space for partarray */
gotpartition = firstpart(inttopart,numel,maxvalue,partarray,offset);
while(gotpartition)
```

```
{
/* do something with partition */
/* get next partition */
gotpartition = nextpart(numel,partarray,offset);
}
```

The `firstpart()` function obtains the first partition simply by zeroing all the elements of the partition (as determined by `offset` and `numel`) and then distributing the integer to be partitioned across the first elements of the array, while taking care that no element exceeds `maxvalue`. Successive partitions are then obtained via repeated calls to `nextpart()`. The generated partitions are ordered in decreasing element value with increasing `part[]` index.

Note that `nextpart()` does not require access to the value of `maxvalue` as `nextpart()` never creates a new partition with element values greater than those in the original partition sent to it. The algorithm of `nextpart()` works by scanning the current partition starting at the highest index and proceeding toward the first element of the partition (i.e., right to left). See the example of the partitions of the number four at the beginning of this appendix for an illustration of the order of the results produced by successive applications of this algorithm. When `nextpart()` finds an element with value greater than one and that is not equal to the preceding element (of index $+ 1$), it attempts a redistribution by decreasing this element value and forming a new "sub-partition" in the cells after it with no element exceeding the new value of the element that was decreased by one.[4] If this redistribution is not possible, `nextpart()` will continue scanning for possible redistribution points until the start of the partition is reached. A successful redistribution will result in a new valid partition. When `nextpart()` cannot perform a redistribution, no more partitions are possible.

[4]This aspect makes the task of generating partitions amicable to implementation by recursive algorithms.

No claims are made with regard to the optimality of the algorithm used in `nextpart()`. In particular, with each execution, `nextpart()` operates without regard to the moves it made during its previous execution. An improved algorithm might take such information into consideration to reduce execution time. However, because `nextpart()`'s memory is confined to those values in the partition sent to it, `nextpart()` does not require the use of other tracking data structures and/or static variables. Thus, `nextpart()` can even be used to "advance" partitions that were generated by other means.

Obviously, it is one thing to state mathematically what needs to be done, but quite another matter to instruct a machine to properly carry out the desired algorithm. The reader might now have some small appreciation for the amount of hidden effort that was required to implement in computer code the full analytic model of this work.

Appendix B

Author Publications

1. M. Shell and J. L. A. Hughes, "Performance of all-optical shared memory architecture packet switch networks using channel grouping under bursty traffic," in *Proceedings of the IEEE Workshop on High Performance Switching and Routing*, Dallas, TX, May 2001, pp. 208–212.

2. M. Shell, M. D. Vaughn, A. Wang, D. J. Blumenthal, P.-J. Rigole, and S. Nilsson, "Experimental demonstration of an all-optical routing node for multihop wavelength routed networks," *IEEE Photonics Technology Letters*, vol. 8, no. 10, pp. 1391–1393, Oct. 1996.

3. M. Shell, M. D. Vaughn, L. Dubertrand, D. J. Blumenthal, P.-J. Rigole, and S. Nilsson, "Multinode demonstration of a multihop wavelength-routed all-optical packet-switched network," in *Proceedings of the Optical Fiber Communications Conference*, vol. 2, Dallas, TX, Feb. 1997, pp. 91–92.

4. M. Shell and D. J. Blumenthal, "Improved crosstalk performance in a 10-wavelength 25 Gbps multichannel laser array transmitter module," in *Digest of the IEEE/LEOS Summer Topical Meetings*, Montreal, Quebec, Canada, Aug. 1997, pp. 64–65.

5. P.-J. Rigole, M. Shell, S. Nilsson, D. J. Blumenthal, and E. Berglind, "Fast wavelength switching in a widely tunable GCSR laser using a pre-distortion technique," in *Proceedings of the Optical Fiber Communications Conference*, vol. 2, Dallas, TX, Feb. 1997, pp. 231–232.

6. D. J. Blumenthal, M. Shell, Q. Gao, M. D. Vaughn, A. Wang, and P.-J. Rigole, "Experimental demonstration of an all-optical routing node for multihop wavelength routed networks," in *Proceedings of the Optical Fiber Communications Conference*, vol. 2, San Jose, CA, Mar. 1996, pp. 108–110.

7. D. J. Blumenthal, J. Laskar, R. Gaudino, S. Han, M. Shell, and M. D. Vaughn, "Fiber-optic links supporting baseband data and subcarrier-multiplexed control channels and the impact of MMIC photonic/microwave interfaces," *IEEE Transactions on Microwave Theory and Techniques*, vol. 45, no. 8, pp. 1443–1452, Aug. 1997.

8. A. Carena, M. D. Vaughn, R. Gaudino, M. Shell, and D. J. Blumenthal, "OPERA: an optical packet experimental routing architecture with label swapping capability," *IEEE/OSA Journal of Lightwave Technology*, vol. 16, no. 12, pp. 2135–2145, Dec. 1998.

9. R. Gaudino, S. Han, M. Shell, M. D. Vaughn, D. J. Blumenthal, and J. Laskar, "A digital-baseband/SCM-control fiber link with novel differential integrated optical transmitter and microwave/optical direct detection receiver," in *Proceedings of the International Topical Meeting on Microwave Photonics*, Essen, Germany, Sept. 1998, pp. 193–196.

10. R. Gaudino, M. Shell, M. Len, G. Desa, C. Juckett, and D. J. Blumenthal, "Experimental demonstration of MOSAIC: a multiwavelength optical subcarrier multiplexed controlled network," in *Proceedings of the Optical Fiber Communications Conference*, vol. 2, San Jose, CA, Feb. 1998, pp. 330–331.

11. R. Gaudino, M. Len, G. Desa, M. Shell, and D. J. Blumenthal, "MOSAIC: a multiwavelength optical subcarrier multiplexed controlled network," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 7, pp. 1270–1285, Sept. 1998.

12. S. Han, R. Gaudino, M. Shell, J. Laskar, and D. J. Blumenthal, "Optimization of RF (MMIC)/optical subcarrier multiplexed communications system," in *IEEE MTT-S International Microwave Symposium Digest*, vol. 2, Denver, CO, June 1997, pp. 979–982.

13. P.-J. Rigole, S. Nilsson, L. Bäckbom, B. Stålnacke, T. Klinga, E. Berglind, B. Stoltz, D. J. Blumenthal, and M. Shell, "Wavelength coverage over 67nm with a GCSR laser: Tuning characteristics and switching speed," in *Proceedings of the 15th IEEE International Semiconductor Laser Conference*, Haifa, Israel, Oct. 1996, pp. 125–126.

14. P.-J. Rigole, S. Nilsson, E. Berglind, D. J. Blumenthal, and M. Shell, "State of the art: Widely tunable lasers," in *Proceedings of the SPIE — In-Plane Semiconductor Lasers: from Ultraviolet to Midinfrared*, San Jose, CA, Feb. 1997, pp. 382–393.

References

[1] K. Fukuchi, T. Kasamatsu, M. Morie, R. Ohhira, T. Ito, K. Sekiya, D. Ogasahara, and T. Ono, "10.92-Tb/s (273 × 40-Gb/s) triple-band/ultra-dense WDM optical-repeatered transmission experiment," in *Proceedings of the Optical Fiber Communications Conference*, vol. 4, Anaheim, CA, Mar. 2001, postdeadline paper PD24-1-3.

[2] S. Bigo *et al.*, "10.2Tbit/s (256×42.7Gbit/s PDM/WDM) transmission over 100km TeraLight^{TM} fiber with 1.28bit/s/Hz spectral efficiency," in *Proceedings of the Optical Fiber Communications Conference*, vol. 4, Anaheim, CA, Mar. 2001, postdeadline paper PD25-1-3.

[3] S. D. Personick, "An engineering perspective on the applications of photonic switching technology," in *Proceedings of the IEEE GLOBECOM Conference*, vol. 2, Atlanta, GA, Nov. 1984, pp. 871–873.

[4] M. Shell, M. D. Vaughn, L. Dubertrand, D. J. Blumenthal, P.-J. Rigole, and S. Nilsson, "Multinode demonstration of a multihop wavelength-routed all-optical packet-switched network," in *Proceedings of the Optical Fiber Communications Conference*, vol. 2, Dallas, TX, Feb. 1997, pp. 91–92.

[5] M. W. Chbat *et al.*, "Toward wide-scale all-optical transparent networking: the ACTS optical pan-european network (OPEN) project," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 7, pp. 1226–1244, Sept. 1998.

[6] L. D. Garrett *et al.*, "The MONET New Jersey network demonstration," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 7, pp. 1199–1219, Sept. 1998.

[7] G. I. Papadimitriou, C. Papazoglou, and A. S. Pomportsis, "Optical switching: Switch fabrics, techniques, and architectures," *IEEE/OSA Journal of Lightwave Technology*, vol. 21, no. 2, pp. 384–405, Feb. 2003.

[8] S. Yao, B. Mukherjee, S. J. B. Yoo, and S. Dixit, "A unified study of contention-resolution schemes in optical packet-switched networks," *IEEE/OSA Journal of Lightwave Technology*, vol. 21, no. 3, pp. 672–683, Mar. 2003.

[9] D. K. Hunter, M. C. Chia, and I. Andonovic, "Buffering in optical packet switches," *IEEE/OSA Journal of Lightwave Technology*, vol. 16, no. 12, pp. 2081–2094, Dec. 1998.

[10] P. D. Bergstrom, Jr., "Markov chain models for all-optical shared memory packet switches," Ph.D. dissertation, Georgia Institute of Technology, Atlanta, Dec. 1998.

[11] P. D. Bergstrom, Jr., M. A. Ingram, A. J. Vernon, J. L. A. Hughes, and P. Tetali, "A markov chain model for an optical shared-memory packet switch," *IEEE Transactions on Communications*, vol. 47, no. 10, pp. 1593–1603, Oct. 1999.

[12] A. Kushwaha, S. K. Bose, and Y. N. Singh, "Analytical modeling for performance studies of an FLBM-based all-optical packet switch," *IEEE Communications Letters*, vol. 5, no. 4, pp. 227–229, Apr. 2001.

[13] Y. N. Singh, A. Kushwaha, and S. K. Bose, "Exact and approximate analytical modeling of an FLBM-based all-optical packet switch," *IEEE/OSA Journal of Lightwave Technology*, vol. 21, no. 3, pp. 719–726, Mar. 2003.

[14] "IBM packet routing switch PRS64G datasheet," IBM Inc., Aug. 2000.

[15] M. Arpaci and J. A. Copeland, "Buffer management for shared-memory ATM switches," *IEEE Communications Surveys & Tutorials*, vol. 3, no. 1, First Quarter 2000. [Online]. Available: http://www.comsoc.org/livepubs/surveys/index.html

[16] A. Huang and S. Knauer, "Starlite: A wideband digital switch," in *Proceedings of the IEEE GLOBECOM Conference*, vol. 1, Atlanta, GA, Nov. 1984, pp. 124–125.

[17] A. Pattavina, *Switching Theory: Architectures and Performance in Broadband ATM Networks*, 1st ed. Chichester, West Sussex, England: John Wiley & Sons, 1998.

[18] G. Bianchi and A. Pattavina, "Architecture and performance of non-blocking ATM switches with shared internal queueing," *Computer Networks and ISDN Systems*, vol. 28, no. 6, pp. 835–853, Apr. 1996.

[19] J. N. Giacopelli, J. J. Hickey, W. S. Marcus, W. D. Sincoskie, and M. Littlewood, "Sunshine: A high performance self-routing broadband packet switch architecture," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 8, pp. 1289–1298, Oct. 1991.

[20] D. J. Blumenthal, J. Laskar, R. Gaudino, S. Han, M. Shell, and M. D. Vaughn, "Fiber-optic links supporting baseband data and subcarrier-multiplexed control channels and the impact of MMIC photonic/microwave interfaces," *IEEE Transactions on Microwave Theory and Techniques*, vol. 45, no. 8, pp. 1443–1452, Aug. 1997.

[21] G. Bendelli, M. Burzio, P. Gambini, and M. Puleo, "Performance assessment of a photonic ATM switch based on a wavelength-controlled fiber loop buffer," in *Proceedings of the Optical Fiber Communications Conference*, vol. 2, San Jose, CA, Mar. 1996, pp. 106–107.

[22] Y. Yamada, K. Sasayama, and K. Habara, "Demonstration of 30 circulations in a transparent optical-loop buffer for 2-channel FDM packets at a data rate of 2.8 Gbit/s," in *Proceedings of the Optical Fiber Communications Conference*, vol. 2, San Jose, CA, Mar. 1996, pp. 107–108.

[23] C. Liu, Z. Dutton, C. H. Behroozi, and L. V. Hau, "Observation of coherent optical information storage in an atomic medium using halted light pulses," *Nature*, vol. 409, no. 6819, pp. 490–493, Jan. 25, 2001.

[24] S. Gianatti and A. Pattavina, "Performance analysis of ATM Banyan networks with shared queueing—part I: Random offered traffic," *IEEE/ACM Transactions on Networking*, vol. 2, no. 4, pp. 398–410, Aug. 1994.

[25] T. Otani, T. Miyazaki, and S. Yamamoto, "40-Gb/s optical 3R regenerator using electroabsorption modulators for optical networks," *IEEE/OSA Journal of Lightwave Technology*, vol. 20, no. 2, pp. 195–200, Feb. 2002.

[26] G. Gavioli and P. Bayvel, "Novel 3R regenerator based on polarization switching in a semiconductor optical amplifier-assisted fiber Sagnac interferometer," *IEEE Photonics Technology Letters*, vol. 15, no. 9, pp. 1261–1263, Sept. 2003.

[27] K. A. McCoy, "A recirculating optical loop for short-term data storage," Ph.D. dissertation, Georgia Institute of Technology, Atlanta, Dec. 1996.

[28] K. L. Hall and K. A. Rauschenbach, "All-optical buffering of 40-Gb/s data packets," *IEEE Photonics Technology Letters*, vol. 10, no. 3, pp. 442–444, Mar. 1998.

[29] Y. Liu, M. T. Hill, N. Calabretta, H. de Waardt, G. D. Khoe, and H. J. S. Dorren, "All-optical buffering in all-optical packet switched cross connects," *IEEE Photonics Technology Letters*, vol. 14, no. 6, pp. 849–851, June 2002.

[30] L. Wang, A. Agarwal, Y. Su, and P. Kumar, "All-optical picosecond-pulse packet buffer based on four-wave mixing loading and intracavity soliton control," *IEEE Journal of Quantum Electronics*, vol. 38, no. 6, pp. 614–619, June 2002.

[31] D. Chiaroni *et al.*, "Physical and logical validation of a network based on all-optical packet switching systems," *IEEE/OSA Journal of Lightwave Technology*, vol. 16, no. 12, pp. 2255–2263, Dec. 1998.

[32] M. Kalyvas, G. Theophilopoulos, C. Bintjas, N. Pleros, A. Stavdas, and H. Avramopoulos, "2×2 all-optical exchange-bypass switch," in *Digest of the IEEE/LEOS Summer Topical Meetings*, Mont Tremblant, Quebec, Canada, July 2002, pp. 121–122.

[33] N. Wada, H. Harai, and F. Kubota, "40 Gbit/s interface, optical code based photonic packet switch prototype," in *Proceedings of the Optical Fiber Communications Conference*, vol. 2, Atlanta, GA, Mar. 2003, pp. 801–802.

[34] M. Y. Jeon, Z. Pan, J. Cao, Y. Bansal, J. Taylor, Z. Wang, V. Akella, K. Okamoto, S. Kamei, J. Pan, and S. J. B. Yoo, "Demonstration of all-optical packet switching routers with optical label swapping and 2R regeneration for scalable optical label switching network applications," *IEEE/OSA Journal of Lightwave Technology*, vol. 21, no. 11, pp. 2723–2733, Nov. 2003.

[35] M. J. Karol, "Shared-memory optical packet (ATM) switch," in *Proceedings of the SPIE Multigigabit Fiber Communications Conference*, San Diego, CA, July 1993, pp. 212–222.

[36] Z. Haas, "'staggering switch': An 'almost-all' optical packet switch," *Electronics Letters*, vol. 28, no. 7, pp. 1576–1577, Aug. 13, 1992.

[37] Z. Haas, "Gbps optical connectivity with the staggering switch," in *Proceedings of the SPIE Multigigabit Fiber Communications Conference*, Boston, MA, Sept. 1992, pp. 180–191.

[38] Z. Haas, "Extensions to the 'staggering switch' architecture," in *Proceedings of the IEEE INFOCOM Conference*, vol. 2, San Francisco, CA, Apr. 1993, pp. 455–463.

[39] Z. Haas, "The staggering switch: An electronically controlled optical packet switch," *IEEE/OSA Journal of Lightwave Technology*, vol. 11, no. 5/6, pp. 925–936, May/June 1993.

[40] Z. Haas, "Comparison of feed-forward and feedback configurations of the staggering switch architecture," in *Proceedings of the SPIE Multigigabit Fiber Communications Conference*, San Diego, CA, July 1993, pp. 244–256.

[41] R. Izmailov and Z. Haas, "Performance evaluation of the optical staggering switch," in *Proceedings of the IEEE GLOBECOM Conference*, vol. 1, Houston, TX, Dec. 1993, pp. 126–132.

[42] T. Szymanski and S. Shaikh, "Markov chain analysis of packet-switched Banyans with arbitrary switch sizes, queue sizes, link multiplicities and speedups," in *Proceedings of the IEEE INFOCOM Conference*, vol. 3, Ottawa, Ontario, Canada, Apr. 1989, pp. 960–971.

[43] S. Liew and K. Lu, "Performance analysis of asymmetric packet switch modules with channel grouping," in *Proceedings of the IEEE INFOCOM Conference*, vol. 2, San Francisco, CA, June 1990, pp. 668–676.

[44] S. C. Liew and K. W. Lu, "Comparison of buffering strategies for asymmetric packet switch modules," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 3, pp. 428–438, Apr. 1991.

[45] S. L. Danielsen, B. Mikkelsen, C. Joergensen, T. Durhuus, and K. E. Stubkjaer, "WDM packet switch architectures and analysis of the influence of tuneable wavelength converters on the performance," *IEEE/OSA Journal of Lightwave Technology*, vol. 15, no. 2, pp. 219–227, Feb. 1997.

[46] M. C. Chia, D. K. Hunter, I. Andonovic, P. Ball, I. Wright, S. P. Ferguson, K. M. Guild, and M. J. O'Mahony, "Packet loss and delay performance of feedback and feed-forward arrayed-waveguide gratings-based optical packet switches with WDM inputs-outputs," *IEEE/OSA Journal of Lightwave Technology*, vol. 19, no. 9, pp. 1241–1254, Sept. 2001.

[47] C.-L. Wu and T.-Y. Feng, "On a class of multistage interconnection networks," *IEEE Transactions on Computers*, vol. C-29, no. 8, pp. 694–702, Aug. 1980.

[48] L. Kleinrock, *Queueing Systems, Volume I: Theory.* New York, NY: John Wiley & Sons, 1975.

[49] R. Nelson, *Probability, Stochastic Processes and Queueing Theory.* New York, NY: Springer-Verlag, 1995.

[50] M. G. Hluchyj and M. J. Karol, "Queueing in space-division packet switching," in *Proceedings of the IEEE INFOCOM Conference*, New Orleans, LA, Mar. 1988, pp. 334–343.

[51] M. G. Hluchyj and M. J. Karol, "Queueing in high-performance packet switching," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1587–1597, Dec. 1988.

[52] M. J. Karol, "Queueing in optical packet switches," in *Proceedings of the SPIE Multigigabit Fiber Communications Conference*, Boston, MA, Sept. 1992, pp. 192–203.

[53] S. C. Liew and K. W. Lu, "A 3-stage interconnection structure for very large packet switches," in *Proceedings of the IEEE SUPERCOMM ICC Conference*, vol. 2, Atlanta, GA, Apr. 1990, pp. 771–777.

[54] A. Y.-M. Lin and J. A. Silvester, "On the performance of an ATM switch with multichannel transmission groups," *IEEE Transactions on Communications*, vol. 41, no. 5, pp. 760–770, May 1993.

[55] J. S. Turner, "Queueing analysis of buffered switching networks," *IEEE Transactions on Communications*, vol. 41, no. 2, pp. 412–420, Feb. 1993.

[56] G. Bianchi and J. S. Turner, "Improved queueing analysis of shared buffer switching networks," *IEEE/ACM Transactions on Networking*, vol. 1, no. 4, pp. 482–490, Aug. 1993.

[57] A. Monterosso and A. Pattavina, "Performance analysis of multistage interconnection networks with shared-buffered switching elements for ATM switching," in *Proceedings of the IEEE INFOCOM Conference*, vol. 1, Florence, Italy, May 1992, pp. 124–131.

[58] A. Pattavina and S. Gianatti, "Performance analysis of ATM Banyan networks with shared queueing—part II: Correlated/unbalanced offered traffic," *IEEE/ACM Transactions on Networking*, vol. 2, no. 4, pp. 411–424, Aug. 1994.

[59] S. Montagna, R. Paglino, and J. F. Meyer, "An integrated approach to evaluating the loss performance of ATM switches," in *Proceedings of the Third Workshop on Performance Modeling and Evaluation of ATM Networks, IFIP Workshop TC6, IFIP Working Groups 6.3 and 6.4*, Ilkley, UK, July 1995, pp. 21/1–10.

[60] S. Fong and S. Singh, "Analytical modelling of shared buffer ATM switches with hot-spot pushout under bursty traffic," in *Proceedings of the IEEE GLOBECOM Conference*, vol. 2, London, U.K., Nov. 1996, pp. 835–839.

[61] S. Fong and S. Singh, "Performance analysis of shared buffer ATM switches with different cell departure models," *International Journal of Communication Systems*, vol. 11, no. 4, pp. 265–274, July/Aug. 1998.

[62] S. Fong and S. Singh, "Queuing analysis of shared-buffer switches with control scheme under bursty traffic," *Computer Communications*, vol. 21, no. 18, pp. 1681–1692, Dec. 1998.

[63] S. Fong and S. Singh, "Analytical modelling of multistage ATM switches with backpressure control schemes," in *Proceedings of the Second IEEE International Workshop on Broadband Switching Systems*, Taiwan, Dec. 1997, pp. 72–83.

[64] M. Saleh and M. Atiquzzaman, "An exact model for analysis of shared buffer delta networks with arbitrary output distribution," in *Proceedings of the Second International Conference on Algorithms and Architectures for Parallel Processing*, Singapore, June 1996, pp. 147–154.

[65] M. Saleh and M. Atiquzzaman, "Accurate modelling of the queueing behavior of shared buffer ATM switches," *International Journal of Communication Systems*, vol. 12, no. 4, pp. 297–308, July/Aug. 1999.

[66] M. Saleh and M. Atiquzzaman, "Analysis of shared buffer switches under non-uniform traffic pattern and global flow control," *Computer Networks*, vol. 34, no. 2, pp. 297–315, Aug. 2000.

[67] J. D. C. Little, "A proof of the queueing formula $l = \lambda w$," *Operations Research*, vol. 9, pp. 383–387, 1961.

[68] W. Whitt, "A review of $l = \lambda w$ and extensions," *Queueing System Theory Applications*, vol. 9, no. 3, pp. 5–68, Oct. 1991.

[69] A. Descloux, "Stochastic models for ATM switching networks," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 3, pp. 450–457, Apr. 1991.

[70] V. J. Friesen and J. W. Wong, "The effect of multiplexing, switching and other factors on the performance of broadband networks," in *Proceedings of the IEEE INFOCOM Conference*, vol. 3, San Francisco, CA, Mar./Apr. 1993, pp. 1194–1203.

[71] T. Szymanski, "An analysis of 'hot-potato' routing in a fiber optic packet switched hypercube," in *Proceedings of the IEEE INFOCOM Conference*, vol. 3, San Francisco, CA, June 1990, pp. 918–925.

[72] Y. Xu and T. Zhang, "Variable shortened-and-punctured Reed-Solomon codes for packet loss protection," *IEEE Transactions on Broadcasting*, vol. 48, no. 3, pp. 237–245, Sept. 2002.

[73] D. J. J. Versfeld, H. C. Ferreira, and A. S. J. Helberg, "A Reed-Solomon decoding algorithm for correcting bursts of erasures in real-time data in packet switched networks," in *Proceedings of the IEEE Information Theory Workshop*, Paris, France, Mar./Apr. 2003, pp. 58–61.

[74] G. E. Andrews, *The Theory of Partitions*. Cambridge, England: Cambridge University Press, 1998.

[75] D. E. Knuth, "Generating all partitions," 2004, pre-fascicle 3B of a draft of sections 7.2.1.4–5 in the proposed volume 4A, "Enumeration and Backtracking," of the *The Art of Computer Programming* series, currently expected to be published in 2007. [Online]. Available: http://www-cs-faculty.stanford.edu/~knuth/taocp.html

[76] L. Euler, "De partitione numerorum," *Novi Commentarii Academiae Scientiarum Imperialis Petropolitanae*, vol. 3, pp. 125–169, 1750, in Latin. [Online]. Available: http://math.dartmouth.edu/~euler/pages/E191.html

[77] G. Hardy and S. A. Ramanujan, "Asymptotic formulae in combinatory analysis," *Proceedings of the London Mathematical Society*, vol. 17, pp. 75–115, 1918.

[78] J. V. Uspensky, "Asymptotic formulae for numerical functions which occur in the theory of partitions," *Bulletin of the Russian Academy of Sciences*, vol. 14, pp. 199–218, 1920.

[79] H. Rademacher, "On the partition function $p(n)$," *Proceedings of the London Mathematical Society*, vol. 43, pp. 241–254, 1937.

[80] M. Abramowitz and I. A. Stegun, Eds., *Handbook of Mathematical Functions*, ser. NBS Applied Mathematics. Washington, D.C.: National Bureau of Standards, 1966, no. 55, ch. 24.

Vita

Michael David Shell was born at Fort Gordon, Georgia on January 18th, 1969. He was the Valedictorian of Curtis Baptist High School's class of 1987. He attended the Georgia Institute of Technology, receiving the B.S.E.E. (magna cum laude) and M.S.E.E. degrees in June of 1991 and in March of 1993, respectively. From 1995 to 1998 he worked in the Optical Communication and Photonic Networks (OCPN) lab of the Georgia Institute of Technology. There, he did research on subsystems for all-optical packet switching nodes. From 1998 to 2004 he worked under the guidance of Dr. Joseph L. A. Hughes and developed new analysis tools and techniques for the optical shared-memory architecture class of packet switches. His research interests include all-optical packet switched networks, high speed opto-electronic interface design, discrete simulation and Markov models for buffered packet switches.

Michael is the author and maintainer of current versions of the IEEEtran and gtphd LaTeX classes, the IEEEtran BibTeX style, as well as the LaTeX testflow diagnostic. His IEEEtran and testflow tools are used by thousands, if not hundreds of thousands, of users worldwide. The IEEEtran BibTeX style and the gtphd LaTeX class were used to typeset this document.

In his spare time, he enjoys tinkering on his '72 Ford, cultivating fruit trees, fishing, philosophizing and practicing the fine art of Unix programming and system administration — sometimes doing more than one at a time.